

Notes on Linear Algebra

Gyubeom Edward Im*

February 24, 2024

Contents

1 Linear Systems	4
1.1 Linear Equation	4
1.2 Linear system	4
1.3 Homogeneous equation	4
1.4 Over-determined system	4
1.5 Under-determined system	5
1.6 Solving Linear System	5
1.7 Linear Combination	5
1.8 Span	5
1.9 From Matrix Equation to Vector Equation	6
1.10 Several Perspectives about Matrix Multiplication	6
1.11 Linear Independence	6
1.12 Linear Dependence	7
1.13 Span and Subspace	7
1.14 Basis of a Subspace	8
1.15 Dimension of Subspace	8
1.16 Column Space of Matrix	8
1.17 Rank of Matrix	8
1.18 Transformation	9
1.19 Linear Transformation	9
1.20 Transformations between Vectors	9
1.21 Matrix of Linear Transformation	9
1.22 Onto and One-To-One	10
2 Least Squares	10
2.1 Inner Product	10
2.2 Properties of Inner Product	10
2.3 Vector Norm	11
2.4 Unit Vector	11
2.5 Distance between Vectors in \mathbb{R}^n	11
2.6 Inner Product and Angle between Vectors	11
2.7 Orthogonal Vectors	11
2.8 Least Square Problem	11
2.9 Normal Equation	12
2.10 Another Derivation of Normal Equation	12
2.11 What If $\mathbf{C} = \mathbf{A}^\top \mathbf{A}$ is NOT Invertible?	12
2.12 Orthogonal Projection Perspective	13
2.13 Orthogonal and Orthonormal Sets	13
2.14 Orthogonal and Orthonormal Basis	13
2.15 Orthogonal Projection $\hat{\mathbf{y}}$ of \mathbf{y} onto Line	13
2.16 Orthogonal Projection $\hat{\mathbf{y}}$ of \mathbf{y} onto Plane	14
2.17 Orthogonal Projection when $\mathbf{y} \in W$	14

*blog: alida.tistory.com, email: criterion.im@gmail.com

2.18	Transformation: Orthogonal Projection	14
2.19	Orthogonal Projection Perspective	15
2.20	Gram-Schmidt Orthogonalization	15
3	Eigenvectors and Eigenvalues	15
3.1	Null Space	16
3.2	Orthogonal Complement	16
3.3	Characteristic Equation	16
3.4	Eigenspace	17
3.5	Diagonalization	17
3.6	Finding \mathbf{V} and \mathbf{D}	17
3.7	Eigendecomposition	17
3.8	Linear Transformation via Eigendecomposition	18
3.9	Change of Basis	18
3.10	Element-wise Scaling	18
3.11	Back to Original Basis	18
3.12	Linear Transformation via \mathbf{A}^k	18
3.13	Geometric Multiplicity and Algebraic Multiplicity	18
4	Singular Value Decomposition	19
4.1	SVD as Sum of Outer Products	19
4.2	Another Perspective of SVD	19
4.3	Computing SVD	20
4.4	Diagonalization of Symmetric Matrices	20
4.5	Spectral Theorem of Symmetric Matrices	20
4.6	Spectral Decomposition	20
4.7	Positive (Semi-)Definite Matrices	21
4.8	Symmetric Positive Definite Matrices	21
4.9	Back to Computing SVD	21
4.10	Eigendecomposition in Machine Learning	21
4.11	Low Rank Approximation of a Matrix	21
4.12	Dimension Reducing Transformation	21
5	Derivative of multi-variable function	22
5.1	Gradient	22
5.2	Jacobian matrix	22
5.2.1	Toy example 1	22
5.2.2	Toy example 2	23
5.3	Hessian matrix	23
5.4	Laplacian	23
5.5	Taylor expansion	24
6	Matrix Algebra	24
6.1	Identity Matrix	24
6.2	Transpose of Matrix	24
6.3	Determinant of Matrix	24
6.4	Inverse Matrix	25
6.5	Trace of Matrix	25
6.6	Diagonal Matrix	25
6.7	Idempotent Matrix	26
7	Matrix Decompositions	26
7.1	LU decomposition	26
7.1.1	PLU decomposition	26
7.1.2	LDU decomposition	27
7.2	Cholesky decomposition	27
7.2.1	Detailed explanation	27
7.3	LDLT decomposition	28
7.4	QR decomposition	28

7.4.1	Detailed explanation	28
7.4.2	QR decomposition on least squares problem	28
7.5	Eigen decomposition	29
7.6	Singular value decomposition	29
7.6.1	Computing SVD	30
7.6.2	Range and nullspace of SVD	30
7.6.3	SVD on under-determined system	30
7.6.4	SVD on over-determined system	30
7.7	Pseudo inverse	30
7.7.1	Pseudo inverse on under-determined system	30
7.7.2	Pseudo inverse on over-determined system	31
7.7.3	SVD of pseudo inverse	31
7.7.4	Full column rank case	32
7.7.5	Full row rank case	32
7.7.6	Rank deficient case	32
7.7.7	QR decomposition of pseudo inverse when singular case	33
7.8	Sherman-Morrison formula	33
7.8.1	Recursive least squares	34
7.9	Matrix inversion lemma	34
7.9.1	Derivation of matrix inversion lemma	35
7.9.2	LDU decomposition	35
7.9.3	UDL decomposition	35
7.9.4	Back to matrix inversion lemma	35
8	Reference	36
9	Revision log	36

1 Linear Systems

1.1 Linear Equation

선형방정식(Linear Equation)은 변수 x_1, \dots, x_n 이 있을 때 다음과 같이 작성할 수 있는 방정식을 의미한다.

$$a_1x_1 + a_2x_2 + \dots + a_nx_n = b \quad (1)$$

이 때, b 는 계수를 의미하고 a_1, \dots, a_n 값들은 실수 또는 복소수의 미지수를 의미한다. 위 식은 다음과 같이 간결하게 작성할 수 있다.

$$\mathbf{a}^T \mathbf{x} = b \quad (2)$$

이 때, $\mathbf{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix}$ 이고 $\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$ 이다.

1.2 Linear system

선형방정식(linear equation)의 집합을 선형시스템(linear system)이라고 한다. n 개의 선형방정식 $\mathbf{a}_1\mathbf{x} = b_1, \dots, \mathbf{a}_n\mathbf{x} = b_n$ 이 있는 경우 이를 다음과 같이 간결하게 선형시스템으로 표현할 수 있다.

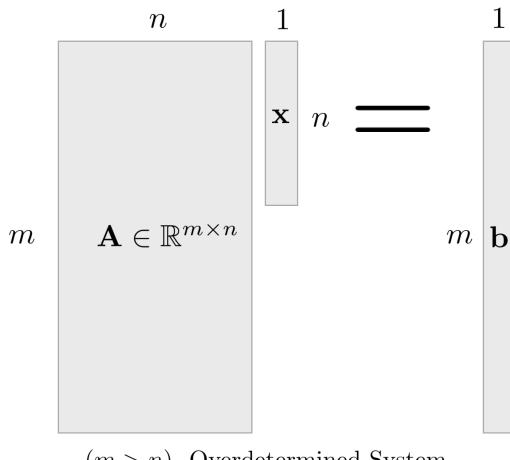
$$\mathbf{Ax} = \mathbf{b} \quad (3)$$

즉, $\mathbf{Ax} = \mathbf{b}$ 형태의 행렬과 벡터의 방정식을 선형시스템이라고 한다. 선형시스템은 다른 말로 비동차방정식이라고도 불린다. 동차방정식, 비동차방정식의 정의는 다음과 같다.

1.3 Homogeneous equation

동차방정식(homogeneous equation)은 $\mathbf{A} \in \mathbb{R}^{n \times m}, \mathbf{x} \in \mathbb{R}^{m \times 1}, \mathbf{b} \in \mathbb{R}^{m \times 1}$ 일 때, $\mathbf{Ax} = 0$ 형태의 시스템을 말한다. 0이 아닌 해가 존재한다. 이와 반대로 $\mathbf{Ax} = \mathbf{b}$ 형태의 방정식을 비동차방정식(non-homogeneous equation)이라고 한다. 비동차방정식은 해가 존재하지 않거나 여러 개 존재한다.

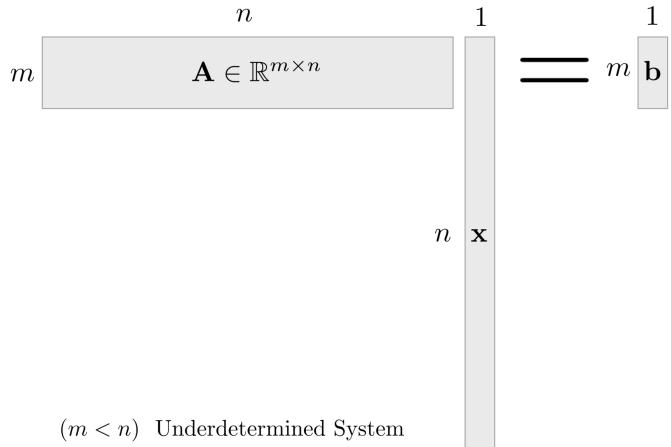
1.4 Over-determined system



Over-determined 시스템은 방정식의 개수가 미지수의 개수보다 많은 경우를 의미한다. $\mathbf{Ax} = \mathbf{b}$ 의 형태에서 $\mathbf{A} \in \mathbb{R}^{m \times n}, \mathbf{x} \in \mathbb{R}^{n \times 1}, \mathbf{b} \in \mathbb{R}^{m \times 1}$ 이라고 했을 때 $m > n$ 인 경우를 의미한다. Over-determined 시스템의 경우 해가 존재하지 않으며 full column rank를 가진다.

$\mathbf{Ax} = \mathbf{b}$ 시스템의 해가 존재하지 않으므로 $\|\mathbf{Ax} - \mathbf{b}\|$ 을 최소화하는 근사해를 구하는 방법을 주로 사용한다.

1.5 Under-determined system



Under-determined 시스템의 경우 방정식의 개수보다 미지수의 개수가 많은 경우를 의미한다. 즉, over-determined 시스템과 반대로 $n > m$ 인 경우를 의미한다. Under-determined 시스템의 경우 무수히 많은 해가 존재하며 full row rank를 가진다.

$\mathbf{Ax} = \mathbf{b}$ 시스템이 무수히 많은 해를 가지므로 $\|\mathbf{x}\|^2$ 가 최소가 되는 해를 구하는 방법을 주로 사용한다.

1.6 Solving Linear System

행렬 \mathbf{A} 의 역행렬이 존재하는 경우 선형시스템은 역행렬을 사용하여 다음과 같이 풀 수 있다.

$$\begin{aligned} \mathbf{Ax} &= \mathbf{b} \\ \mathbf{A}^{-1}\mathbf{Ax} &= \mathbf{A}^{-1}\mathbf{b} \\ \mathbf{Ix} &= \mathbf{A}^{-1}\mathbf{b} \\ \mathbf{x} &= \mathbf{A}^{-1}\mathbf{b} \end{aligned} \tag{4}$$

그러나, 행렬 \mathbf{A} 의 판별식 $\det \mathbf{A} = 0$ 인 경우 역행렬이 존재하지 않게되고 위와 같이 문제를 풀 수 없다. 이런 경우 선형시스템은 해가 존재하지 않거나 무수히 많은 해가 존재한다.

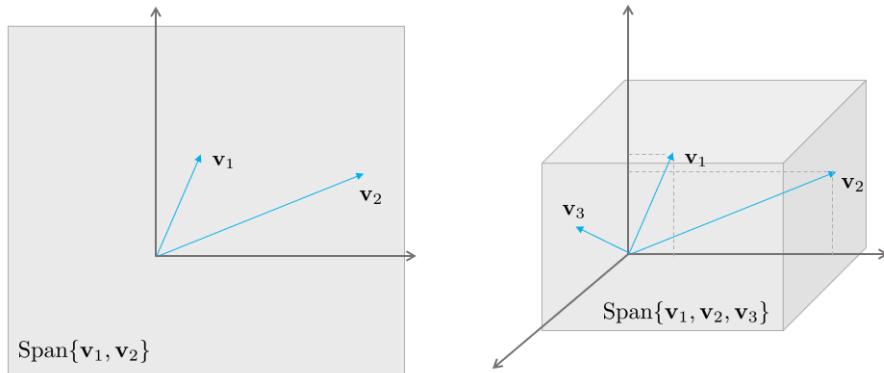
1.7 Linear Combination

여러 벡터 $\mathbf{v}_1, \dots, \mathbf{v}_n \in \mathbb{R}^n$ 이 있을 때 스칼라 값 c_1, \dots, c_n 에 대하여

$$c_1\mathbf{v}_1 + \dots + c_n\mathbf{v}_n \tag{5}$$

을 벡터 $\mathbf{v}_1, \dots, \mathbf{v}_n$ 의 가중치 계수 c_1, \dots, c_n 에 대한 **선형결합 (Linear Combination)**이라고 한다. 이 때 가중치 계수 c_1, \dots, c_n 는 0을 포함한 실수 값을 가진다.

1.8 Span



주어진 여러 벡터 $\mathbf{v}_1, \dots, \mathbf{v}_n \in \mathbb{R}^n$ 에 대해 $\text{Span}\{\mathbf{v}_1, \dots, \mathbf{v}_n\}$ 은 모든 $\mathbf{v}_1, \dots, \mathbf{v}_n$ 에 대한 선형결합의 집합을 의미한다. 즉, $\text{Span}\{\mathbf{v}_1, \dots, \mathbf{v}_n\}$ 은 다음과 같이 쓸 수 있는 모든 벡터들의 집합이다.

$$c_1\mathbf{v}_1 + \dots + c_n\mathbf{v}_n \quad (6)$$

이는 또한 $\mathbf{v}_1, \dots, \mathbf{v}_n$ 에 의해 span 된 \mathbb{R}^n 공간 상의 subset이라고도 불린다.

1.9 From Matrix Equation to Vector Equation

$\mathbf{Ax} = \mathbf{b}$ 와 같은 선형 시스템을 다음과 같이 열벡터 \mathbf{a}_i 를 기준으로 펼쳐보면

$$[\mathbf{a}_1 \quad \mathbf{a}_2 \quad \cdots \quad \mathbf{a}_n] \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \mathbf{b} \quad (7)$$

로 나타낼 수 있고 이를 다시 표현하면

$$\mathbf{a}_1x_1 + \mathbf{a}_2x_2 + \cdots + \mathbf{a}_nx_n = \mathbf{b} \quad (8)$$

와 같이 열벡터들의 선형결합으로 표현할 수 있게 된다. 만약 \mathbf{b} 가 $\text{Span}\{\mathbf{a}_1, \dots, \mathbf{a}_n\}$ 에 포함되어 있다면 이들의 선형결합으로 표현할 수 있으므로 해가 존재한다. 따라서 $\mathbf{b} \in \text{Span}\{\mathbf{a}_1, \dots, \mathbf{a}_n\}$ 일 때 해가 존재한다.

1.10 Several Perspectives about Matrix Multiplication

선형시스템 $\mathbf{Ax} = \mathbf{b}$ 가 있을 때 이는 곧 \mathbf{A} 의 열벡터들의 선형결합으로 표현할 수 있다.

$$\mathbf{Ax} = [\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n] \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \mathbf{a}_1x_1 + \mathbf{a}_2x_2 + \cdots + \mathbf{a}_nx_n = \mathbf{b} \quad (9)$$

만약 선형시스템에 전치행렬을 적용하여 $\mathbf{x}^\top \mathbf{A}^\top = \mathbf{b}^\top$ 가 되면

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_n \end{bmatrix} = \mathbf{a}_1x_1 + \mathbf{a}_2x_2 + \cdots + \mathbf{a}_nx_n = \mathbf{b} \quad (10)$$

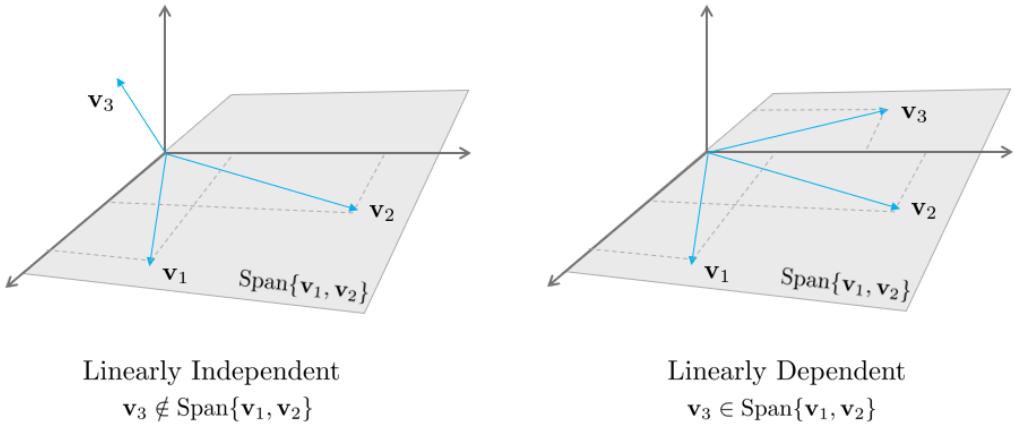
\mathbf{b}^\top 는 곧 \mathbf{A}^\top 의 행벡터(Row Vector)들의 선형결합으로 표현된다.

또한 두 벡터의 곱 $\mathbf{ab}^\top = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \begin{bmatrix} b_1 & \cdots & b_n \end{bmatrix}$ 의 경우 rank1 outer product로 볼 수 있다. 즉, $[\mathbf{a} \ \mathbf{c}] \begin{bmatrix} \mathbf{b} \\ \mathbf{d} \end{bmatrix}$ 의 경우 $\mathbf{ab} + \mathbf{cd}$ 와 같이 벡터곱을 스칼라 곱과 같이 생각할 수 있다.

1.11 Linear Independence

벡터 집합 $\mathbf{v}_1, \dots, \mathbf{v}_n \in \mathbb{R}^n$ 가 주어졌을 때, 이들 중 부분 벡터들의 집합 $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{j-1}\}$ 이 선형결합을 통해 특정 벡터 $\mathbf{v}_j, j = 1, \dots, n$ 를 표현할 수 있는지 검사한다.

$$\mathbf{v}_j \in \text{Span}\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{j-1}\} \quad \text{for some } j = 1, \dots, n? \quad (11)$$



Linearly Independent
 $v_3 \notin \text{Span}\{v_1, v_2\}$

Linearly Dependent
 $v_3 \in \text{Span}\{v_1, v_2\}$

만약 v_j 가 선형결합으로 표현이 된다면 v_1, \dots, v_n 는 선형의존 (Linearly Dependent)이다. 만약, v_j 가 표현되지 않는다면 v_1, \dots, v_n 는 선형독립 (Linearly Independent)이다.

만약 $x_1v_1 + x_2v_2 + \dots + x_nv_n = \mathbf{0}$ 같은 동차(homogeneous) 선형방정식이 있다고 하면

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (12)$$

과 같은 자명해가 존재한다. 이 때, v_1, \dots, v_n 가 선형독립이면 자명해 이외에 해는 존재하지 않는다. 하지만, v_1, \dots, v_n 가 선형의존이면 선형시스템은 자명해 이외에 다른 해가 존재한다.

자명해 이외에 다른 해가 존재하는 선형의존(Linearly Dependent) 경우 대해서 생각해보면 예를 들어 \mathbf{A} 행렬이 다음과 같이 5개의 열을 가진 행렬이라고 했을 때

$$\mathbf{A} = [\mathbf{a}_1 \ \mathbf{a}_2 \ \mathbf{a}_3 \ \mathbf{a}_4 \ \mathbf{a}_5] \quad (13)$$

위 열벡터(Column Vector)들 중 최소한 두 개 이상의 벡터가 선형결합되어야 동차방정식 $\mathbf{Ax} = \mathbf{0}$ 의 해를 만족할 수 있다. 예를 들어 \mathbf{a}_2x_2 성분이 0이 아닌 경우 이를 다시 영벡터로 만들기 위해서는 다른 1,3,4,5 열벡터들의 선형결합이 $-\mathbf{a}_2x_2$ 의 값을 만들어야 한다. 이는 곧 \mathbf{a}_2x_2 값을 다른 열벡터들의 선형결합으로 표현할 수 있다는 말과 동치이므로 선형의존인 경우 어떤 하나의 벡터가 다른 벡터들의 선형결합으로 표현될 수 있음을 의미한다. 이를 수식으로 표현하면 다음과 같다.

$$\begin{aligned} \mathbf{a}_jx_j &= -\mathbf{a}_1x_1 - \dots - \mathbf{a}_{j-1}x_{j-1} \\ \mathbf{a}_j &= -\frac{x_1}{x_j}\mathbf{a}_1 - \dots - \frac{x_{j-1}}{x_j}\mathbf{a}_{j-1} \in \text{Span}\{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_{j-1}\} \end{aligned} \quad (14)$$

1.12 Linear Dependence

행렬 \mathbf{A} 의 열벡터 $\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n$ 이 선형의존(Linearly Dependent)인 경우 해당 열벡터들은 Span의 차원을 늘리지 않는다. 만약 $\mathbf{A} \in \mathbb{R}^{3 \times 3}$ 이고 $\mathbf{a}_3 \in \text{Span}\{\mathbf{a}_1, \mathbf{a}_2\}$ 인 경우

$$\text{Span}\{\mathbf{a}_1, \mathbf{a}_2\} = \text{Span}\{\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3\} \quad (15)$$

만약 $\mathbf{a}_3 = d_1\mathbf{a}_1 + d_2\mathbf{a}_2$ 와 같이 선형결합으로 표현이 가능한 경우, $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3$ 는 다음과 같이 작성할 수 있다.

$$c_1\mathbf{a}_1 + c_2\mathbf{a}_2 + c_3\mathbf{a}_3 = (c_1 + d_1)\mathbf{a}_1 + (c_1 + d_1)\mathbf{a}_2 \quad (16)$$

1.13 Span and Subspace

\mathbb{R}^n 공간의 부분공간(Subspace) H 는 \mathbb{R}^n 의 부분집합들의 선형결합에 대해 닫혀 있는 공간을 의미한다. 즉, 두 벡터 $\mathbf{u}_1, \mathbf{u}_2 \in H$ 일 때, 어떠한 스칼라 값 c, d 에 대하여 $c\mathbf{u}_1 + d\mathbf{u}_2 \in H$ 일 때 H 를 부분공간이라고 한다.

$\text{Span}\{\mathbf{a}_1, \dots, \mathbf{a}_n\}$ 으로 형성된 공간은 항상 부분공간이다. 만약 $\mathbf{u}_1 = x_1\mathbf{a}_1 + \dots + x_n\mathbf{a}_n$ 이고 $\mathbf{u}_2 = y_1\mathbf{a}_1 + \dots + y_n\mathbf{a}_n$ 일 때

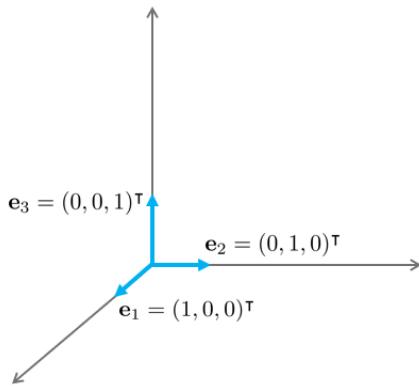
$$\begin{aligned} c\mathbf{u}_1 + d\mathbf{u}_2 &= c(x_1\mathbf{a}_1 + \cdots + x_n\mathbf{a}_n) + d(y_1\mathbf{a}_1 + \cdots + y_n\mathbf{a}_n) \\ &= (cx_1 + dy_1)\mathbf{a}_1 + \cdots + (cx_n + dy_n)\mathbf{a}_n \end{aligned} \quad (17)$$

과 같이 선형결합으로 나타낼 수 있고 이는 임의의 값 c, d 에 대해서 닫혀 있음을 의미한다. 따라서 부분 공간은 항상 $\text{Span } \{\mathbf{a}_1, \dots, \mathbf{a}_n\}$ 으로 표현된다.

1.14 Basis of a Subspace

부분공간 H 의 **기저(basis)**는 다음을 만족하는 벡터들의 집합을 의미한다.

1. 부분공간 H 를 모두 Span 할 수 있어야 한다.
2. 벡터들 간 선형독립이어야 한다.



3차원 공간 \mathbb{R}^3 의 경우 기저벡터는 3개가 존재하고 $\mathbf{e}_1 = [1 \ 0 \ 0]^\top, \mathbf{e}_2 = [0 \ 1 \ 0]^\top, \mathbf{e}_3 = [0 \ 0 \ 1]^\top$ 일 때, 이를 **표준기저벡터(Standard Basis Vector)**라고 한다.

1.15 Dimension of Subspace

하나의 부분공간 H 를 표현할 수 있는 기저는 유일하지 않다. 하지만 여러개의 기저를 통해서 표현할 수 있는 부분공간의 차원(Dimension)은 유일하다. **부분공간의 차원은 기저벡터의 개수와 동일하다.**

1.16 Column Space of Matrix

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_{1,col} & \mathbf{a}_{i,col} & \mathbf{a}_{m,col} \\ \begin{matrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nm} \end{matrix} \end{bmatrix} \in \mathbb{R}^{n \times m} \quad \text{Col } \mathbf{A} = \text{Span}\{\mathbf{a}_{1,col}, \mathbf{a}_{i,col}, \mathbf{a}_{m,col}\}$$

행렬 \mathbf{A} 의 열공간(Column Space)이란 \mathbf{A} 의 열벡터로 인해 Span 된 부분공간을 의미한다. 일반적으로 $\text{Col } \mathbf{A}$ 라고 표기한다.

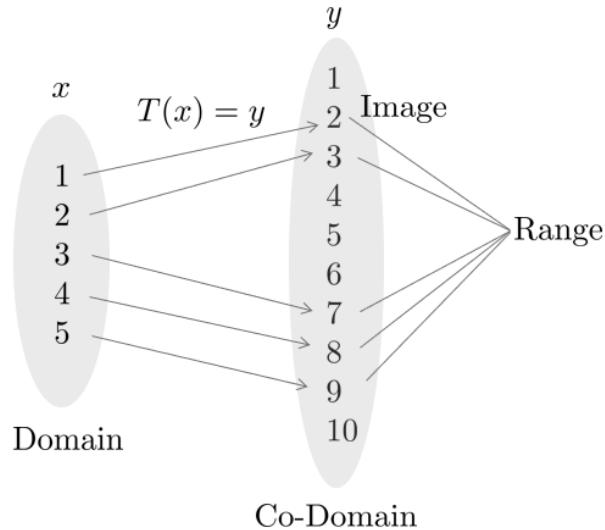
$$\text{Col } \mathbf{A} = \text{Span}\left\{\begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}\right\} \quad (18)$$

1.17 Rank of Matrix

행렬 \mathbf{A} 의 rank란 \mathbf{A} 의 열벡터들의 차원을 의미한다.

$$\text{rank } \mathbf{A} = \dim \text{Col } \mathbf{A} \quad (19)$$

1.18 Transformation



변환(Transformation), 함수(Function), 매핑(Mapping) T 은 입력 x 를 출력 y 로 매핑해주는 것을 의미한다.

$$T : x \mapsto y \quad (20)$$

이 때 입력 x 에 의해 매핑되는 출력 y 는 유일하게 결정된다. **Domain**(정의역)이란 입력 x 의 모든 가능한 집합을 의미한다. **Co-Domain**(공역)이란 출력 y 의 모든 가능한 집합을 의미한다. **Image**란 주어진 입력 x 에 대해 매핑된 출력 y 를 의미한다. **Range**(치역)란 Domain내에 있는 입력 x 들에 의해 매핑된 모든 출력 y 의 집합을 의미한다.

1.19 Linear Transformation

변환 T 는 다음과 같은 경우에 선형변환(Linear Transformation)이라고 한다.

$$T(c\mathbf{u} + d\mathbf{v}) = cT(\mathbf{u}) + dT(\mathbf{v}) \quad (21)$$

for all \mathbf{u}, \mathbf{v} in the domain of T and for all scalars c and d .

1.20 Transformations between Vectors

$T : \mathbf{x} \in \mathbb{R}^n \mapsto \mathbf{y} \in \mathbb{R}^m$ 은 n 차원의 벡터를 m 차원의 벡터로 매핑하는 연산을 의미한다. 예를 들면

$$T : \mathbf{x} \in \mathbb{R}^3 \mapsto \mathbf{y} \in \mathbb{R}^2$$

$$\mathbf{x} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \in \mathbb{R}^3 \mapsto \mathbf{y} = T(\mathbf{x}) = \begin{bmatrix} 4 \\ 5 \end{bmatrix} \in \mathbb{R}^2 \quad (22)$$

1.21 Matrix of Linear Transformation

변환 $T : \mathbb{R}^n \mapsto \mathbb{R}^m$ 을 선형변환이라고 가정하면 T 는 항상 행렬과 벡터의 곱으로 표현할 수 있다. 즉,

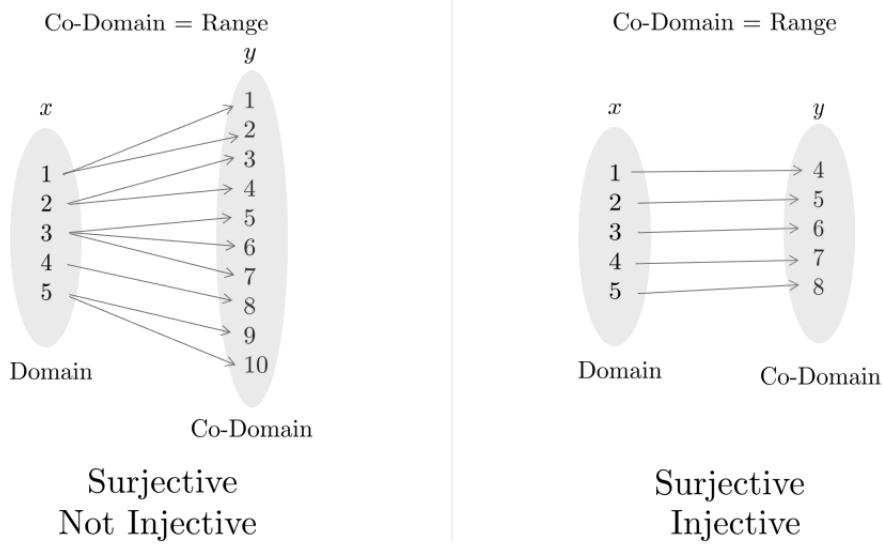
$$T(\mathbf{x}) = \mathbf{Ax} \quad \text{for all } \mathbf{x} \in \mathbb{R}^n \quad (23)$$

행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 인 경우 \mathbf{A} 의 j 번째 열 \mathbf{a}_j 는 벡터 $T(\mathbf{e}_j)$ 와 같다. 이 때 \mathbf{e}_j 는 항등행렬 $\mathbf{I} \in \mathbb{R}^{n \times n}$ 의 j 번째 열벡터이다.

$$\mathbf{A} = [T(\mathbf{e}_1) \ \cdots \ T(\mathbf{e}_n)] \quad (24)$$

이러한 행렬 \mathbf{A} 를 선형변환 T 의 표준행렬(Standard Matrix)이라고 부른다.

1.22 Onto and One-To-One



Onto는 전사함수(Surjective)라고도 불리며 공역이 치역과 같은 경우를 의미한다. 이는 Co-Domain의 모든 원소들이 사영된 것을 의미한다.

$$\text{Surjective: Co-Domain} = \text{Range} \quad (25)$$

One-To-One은 일대일함수(Injective)라고도 불리며 정의역의 원소와 공역의 원소가 하나씩 대응되는 함수를 의미한다.

2 Least Squares

최소제곱법(Least Square)는 방정식의 개수가 미지수의 개수보다 많은 Over-determined 선형시스템에서 사용하는 방법 중 하나이다. Over-determined 선형시스템 $\mathbf{Ax} = \mathbf{b}$ 의 경우 일반적으로 해가 존재하지 않는다. 이런 경우 일반적으로 $\|\mathbf{Ax} - \mathbf{b}\|^2$ 가 최소가 되는 근사해를 구할 수 있다.

2.1 Inner Product

벡터 $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ 에 대해 이를 각각 $n \times 1$ 의 행렬로 생각할 수 있다. 그렇다면 \mathbf{u}^\top 는 $1 \times n$ 의 행렬로 볼 수 있고 행렬곱 $\mathbf{u}^\top \mathbf{v}$ 는 1×1 의 행렬이 된다. 그리고 1×1 행렬은 스칼라값으로 표시할 수 있다.

이 때, $\mathbf{u}^\top \mathbf{v}$ 에 의해 계산된 값을 \mathbf{u}, \mathbf{v} 의 내적(Inner Product, Dot Product)라고 한다. 이는 $\mathbf{u} \cdot \mathbf{v}$ 로 표기할 수 있다.

2.2 Properties of Inner Product

벡터 $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathbb{R}^n$ 이고 c 를 스칼라 값이라고 할 때 내적은 다음과 같은 성질을 만족한다.

1. $\mathbf{u} \cdot \mathbf{v} = \mathbf{v} \cdot \mathbf{u}$
2. $(\mathbf{u} + \mathbf{v}) \cdot \mathbf{w} = \mathbf{u} \cdot \mathbf{w} + \mathbf{v} \cdot \mathbf{w}$
3. $(c\mathbf{u}) \cdot \mathbf{v} = c(\mathbf{u} \cdot \mathbf{v}) = \mathbf{u} \cdot (c\mathbf{v})$
4. $\mathbf{u} \cdot \mathbf{u} \geq 0$ and $\mathbf{u} \cdot \mathbf{u} = 0$ iff $\mathbf{u} = 0$

위에서 2,3번 성질을 조합하면 다음과 같은 법칙을 만들 수 있다.

$$(c_1\mathbf{u}_1 + \cdots + c_n\mathbf{u}_n) \cdot \mathbf{w} = c_1(\mathbf{u}_1 \cdot \mathbf{w}) + \cdots + c_n(\mathbf{u}_n \cdot \mathbf{w}) \quad (26)$$

위를 통해 내적이라는 연산은 선형변환이라는 것을 알 수 있다.

2.3 Vector Norm

벡터 $\mathbf{v} \in \mathbb{R}^n$ 에 대해 벡터의 놈(Norm)은 0이 아닌 $\|\mathbf{v}\| = \sqrt{\mathbf{v} \cdot \mathbf{v}}$ 로 표기하며 벡터의 길이를 의미한다.

$$\|\mathbf{v}\| = \sqrt{\mathbf{v} \cdot \mathbf{v}} = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2} \quad (27)$$

2차원 벡터 $\mathbf{v} \in \mathbb{R}^2$ 가 있을 때 $\mathbf{v} = \begin{bmatrix} a \\ b \end{bmatrix}$ 라고 하면 $\|\mathbf{v}\|$ 는 원점으로부터 \mathbf{v} 좌표까지의 거리가 된다.

$$\|\mathbf{v}\| = \sqrt{a^2 + b^2} \quad (28)$$

모든 스칼라 값 c 에 대해 $c\mathbf{v}$ 의 길이는 \mathbf{v} 의 길이를 $|c|$ 배 한 것을 의미한다.

$$\|c\mathbf{v}\| = |c| \|\mathbf{v}\| \quad (29)$$

2.4 Unit Vector

길이가 1인 벡터를 단위벡터(Unit Vector)라고 한다. 벡터의 길이를 1로 맞추는 작업을 정규화(Normalization)라고 하는데 주어진 벡터 \mathbf{v} 가 있을 때 단위벡터 $\mathbf{u} = \frac{1}{\|\mathbf{v}\|} \mathbf{v}$ 가 된다. \mathbf{u} 벡터는 \mathbf{v} 벡터와 방향은 같지만 크기가 1인 벡터이다.

2.5 Distance between Vectors in \mathbb{R}^n

두 벡터 $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ 이 있을 때 두 벡터의 거리는 $\text{dist}(\mathbf{u}, \mathbf{v})$ 로 나타내며 이는 $\mathbf{u} - \mathbf{v}$ 벡터의 길이를 의미한다.

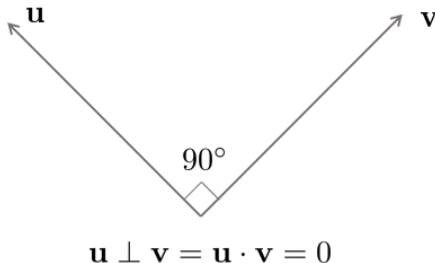
$$\text{dist}(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\| \quad (30)$$

2.6 Inner Product and Angle between Vectors

두 벡터 \mathbf{u}, \mathbf{v} 의 내적은 다음과 같이 놈과 각도를 통해 표현할 수 있다.

$$\mathbf{u} \cdot \mathbf{v} = \|\mathbf{u}\| \|\mathbf{v}\| \cos \theta \quad (31)$$

2.7 Orthogonal Vectors



두 벡터 $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ 가 있을 때 둘이 수직이려면 두 벡터의 내적이 0이어야 한다.

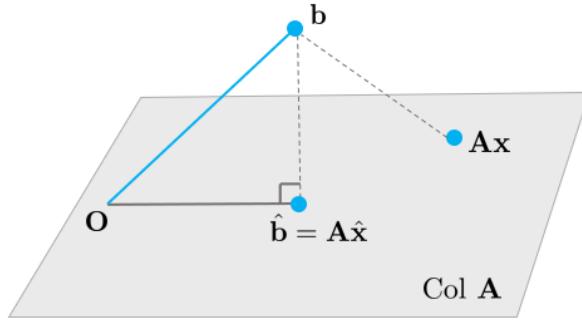
$$\mathbf{u} \cdot \mathbf{v} = \|\mathbf{u}\| \|\mathbf{v}\| \cos \theta = 0 \quad (32)$$

0이 아닌 두 벡터 \mathbf{u}, \mathbf{v} 의 내적이 0이려면 $\cos \theta$ 값이 0이어야 하고 $\theta = 90^\circ$ 일 때 $\cos \theta$ 값은 0이 된다.

2.8 Least Square Problem

$\mathbf{A} \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{R}^n, m \ll n$ 과 같이 주어진 Over-Determined 시스템 $\mathbf{Ax} = \mathbf{b}$ 가 있을 때 여러의 제곱합 $\|\mathbf{b} - \mathbf{Ax}\|$ 을 최소화하는 최적의 모델 파라미터를 찾는 것이 목적이 된다. 이 때 최소제곱법의 근사해 $\hat{\mathbf{x}}$ 는 다음과 같다.

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x}} \|\mathbf{b} - \mathbf{Ax}\| \quad (33)$$



$$Ax = b \Rightarrow A\hat{x} = \hat{b}$$

최소제곱법의 중요한 포인트 중 하나는 어떤 x 파라미터를 설정하던지 벡터 Ax 는 반드시 $\text{Col } A$ 안에 위치한다는 것이다. 따라서 **최소제곱법은 $\text{Col } A$ 와 b 의 거리가 최소가 되는 x 를 찾는 문제가 된다.**

$\hat{b} = A\hat{x}$ 를 만족하는 근사해 \hat{x} 는 $\text{Col } A$ 에서 b 벡터와 가장 가까운 모든 포인트들의 집합을 의미한다. 따라서 b 는 다른 어떤 Ax 보다도 \hat{b} 와 가장 가깝게 된다. 기하학적으로 이를 만족하기 위해서는 벡터 $b - A\hat{x}$ 가 $\text{Col } A$ 와 수직이어야 한다.

$$b - A\hat{x} \perp (x_1 a_1 + x_2 a_2 + \cdots + x_n a_n) \quad \text{for any vector } x. \quad (34)$$

이는 곧 다음과 동일하다.

$$\begin{aligned} (b - A\hat{x}) \perp a_1 &\rightarrow a_1^T(b - A\hat{x}) \\ (b - A\hat{x}) \perp a_2 &\rightarrow a_2^T(b - A\hat{x}) \\ (b - A\hat{x}) \perp a_3 &\rightarrow a_3^T(b - A\hat{x}) \\ \therefore A^T(b - A\hat{x}) &= 0 \end{aligned} \quad (35)$$

2.9 Normal Equation

$Ax \approx b$ 를 만족하는 최소제곱법의 근사해는 다음과 같다.

$$A^T A \hat{x} = A^T b \quad (36)$$

위 식을 **정규방정식(Normal Equation)**이라고 부른다. 이는 $C = A^T A \in \mathbb{R}^{n \times n}$, $d = A^T b \in \mathbb{R}^n$ 일 때 $Cx = d$ 와 같은 선형시스템으로 생각할 수 있다. 이 선형시스템의 해를 구하면 다음과 같다.

$$\hat{x} = (A^T A)^{-1} A^T b \quad (37)$$

2.10 Another Derivation of Normal Equation

근사해 $\hat{x} = \arg \min_x \|b - Ax\| = \arg \min_x \|b - Ax\|^2$ 와 같이 제곱을 최소화하는 문제로 표현해도 동일한 문제가 된다.

$$\arg \min_x (b - Ax)^T (b - Ax) = b^T b - x^T A^T b - b^T A x + x^T A^T A x \quad (38)$$

위 식을 x 에 대해서 미분하고 정리하면 다음과 같다.

$$-A^T b - A^T b + 2A^T A x = 0 \Leftrightarrow A^T A x = A^T b \quad (39)$$

이 때 $A^T A$ 가 역행렬이 존재한다면 다음과 같이 해를 구할 수 있다.

$$x = (A^T A)^{-1} A^T b \quad (40)$$

2.11 What If $C = A^T A$ is NOT Invertible?

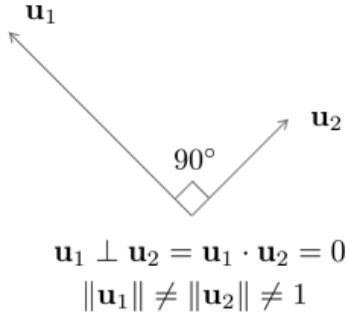
행렬 $C = A^T A$ 의 역행렬이 존재하지 않는 경우 시스템은 해가 없거나 무수히 많은 해를 가지고 있다. 하지만 정규방정식은 항상 해를 가지고 있으므로 해가 없는 상황은 존재하지 않고 실제로는 무수히 많은 해를 가지고 있다. C 가 역행렬을 구할 수 없는 경우는 오직 $\text{Col } A$ 가 선형의존일 경우에 발생한다. 하지만, 일반적으로 C 는 대부분의 경우 역행렬이 존재한다.

2.12 Orthogonal Projection Perspective

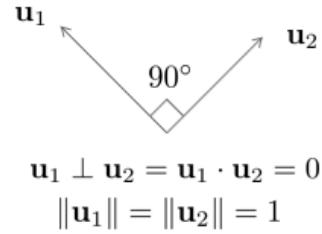
행렬 $C = A^T A$ 가 있을 때 b 점에서 Col A 공간으로 프로젝션하면 다음과 같다.

$$\hat{b} = f(b) = Ax = A(A^T A)^{-1} A^T b \quad (41)$$

2.13 Orthogonal and Orthonormal Sets



Orthogonal Set



Orthonormal Set

벡터들의 집합 $\mathbf{u}_1, \dots, \mathbf{u}_n \in \mathbb{R}^n$ 가 있을 때 모든 벡터 쌍들이 $\mathbf{u}_i \cdot \mathbf{u}_j = 0, i \neq j$ 를 만족하면 해당 집합은 **직교(Orthogonal)**하다고 말한다.

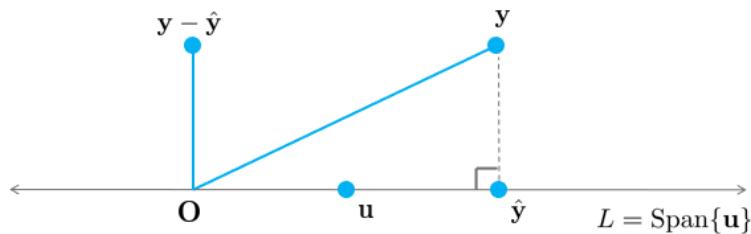
벡터들의 집합 $\mathbf{u}_1, \dots, \mathbf{u}_n \in \mathbb{R}^n$ 가 있을 때 모든 직교 집합들이 단위벡터인 경우 **정규직교(Orthonormal)**하다고 말한다.

직교벡터와 정규직교벡터의 집합은 **항상 선형독립이다**.

2.14 Orthogonal and Orthonormal Basis

기저벡터 $\mathbf{u}_1, \dots, \mathbf{u}_n$ 이 p차원의 부분공간 $W \in \mathbb{R}^n$ 에 있다고 할 때 Gram-Schmidt 프로세스와 QR decomposition을 사용하면 직교기저벡터를 만들 수 있다. 부분공간 W 에 대해 직교기저 벡터 $\mathbf{u}_1, \dots, \mathbf{u}_n$ 이 주어져 있다고 했을 때 $y \in \mathbb{R}^n$ 을 부분공간 W 위로 프로젝션시킨다.

2.15 Orthogonal Projection \hat{y} of y onto Line



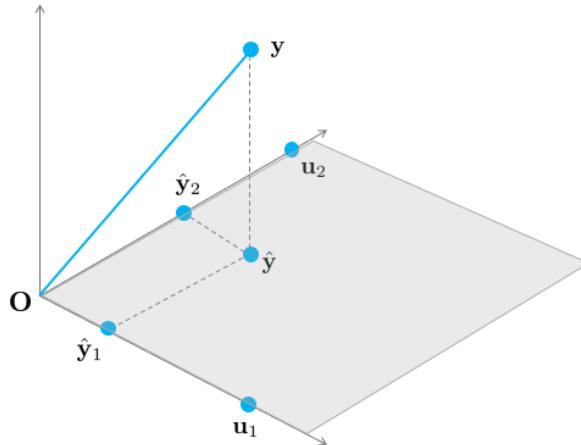
1차원 부분공간 $L = \text{Span}\{\mathbf{u}\}$ 위로 y 를 프로젝션하여 \hat{y} 를 구하면 다음과 같다.

$$\hat{y} = \text{proj}_L y = \frac{y \cdot \mathbf{u}}{\mathbf{u} \cdot \mathbf{u}} \mathbf{u} \quad (42)$$

가 된다. 만약 \mathbf{u} 가 단위벡터이면 다음과 같다.

$$\hat{y} = \text{proj}_L y = (y \cdot \mathbf{u}) \mathbf{u} \quad (43)$$

2.16 Orthogonal Projection \hat{y} of y onto Plane



2차원 부분공간 $W = \text{Span}\{\mathbf{u}_1, \mathbf{u}_2\}$ 위로 \mathbf{y} 를 프로젝션하여 $\hat{\mathbf{y}}$ 를 구하면 다음과 같다.

$$\hat{\mathbf{y}} = \text{proj}_L \mathbf{y} = \frac{\mathbf{y} \cdot \mathbf{u}_1}{\mathbf{u}_1 \cdot \mathbf{u}_1} \mathbf{u}_1 + \frac{\mathbf{y} \cdot \mathbf{u}_2}{\mathbf{u}_2 \cdot \mathbf{u}_2} \mathbf{u}_2 \quad (44)$$

만약 $\mathbf{u}_1, \mathbf{u}_2$ 가 단위벡터이면 다음과 같다.

$$\hat{\mathbf{y}} = \text{proj}_L \mathbf{y} = (\mathbf{y} \cdot \mathbf{u}_1) \mathbf{u}_1 + (\mathbf{y} \cdot \mathbf{u}_2) \mathbf{u}_2 \quad (45)$$

프로젝션은 각각 직교기저벡터에 독립적으로 적용된다.

2.17 Orthogonal Projection when $\mathbf{y} \in W$

만약 2차원 부분공간 $W = \text{Span}\{\mathbf{u}_1, \mathbf{u}_2\}$ 에 \mathbf{y} 가 포함되어 있다고 하면 프로젝션된 벡터 $\hat{\mathbf{y}}$ 는 다음과 같이 구할 수 있다.

$$\hat{\mathbf{y}} = \text{proj}_L \mathbf{y} = \mathbf{y} = \frac{\mathbf{y} \cdot \mathbf{u}_1}{\mathbf{u}_1 \cdot \mathbf{u}_1} \mathbf{u}_1 + \frac{\mathbf{y} \cdot \mathbf{u}_2}{\mathbf{u}_2 \cdot \mathbf{u}_2} \mathbf{u}_2 \quad (46)$$

만약 $\mathbf{u}_1, \mathbf{u}_2$ 가 단위벡터이면 다음과 같다.

$$\hat{\mathbf{y}} = \text{proj}_L \mathbf{y} = \mathbf{y} = (\mathbf{y} \cdot \mathbf{u}_1) \mathbf{u}_1 + (\mathbf{y} \cdot \mathbf{u}_2) \mathbf{u}_2 \quad (47)$$

이는 \mathbf{y} 가 부분공간 W 에 포함되어 있지 않은 경우와 동일하다.

2.18 Transformation: Orthogonal Projection

부분공간 W 의 정규직교기저벡터 $\mathbf{u}_1, \mathbf{u}_2$ 가 있고 \mathbf{b} 를 부분공간 W 에 프로젝션시킨 점 $\hat{\mathbf{b}}$ 의 변환을 생각해보면

$$\begin{aligned} \hat{\mathbf{b}} &= f(\mathbf{b}) = (\mathbf{b} \cdot \mathbf{u}_1) \mathbf{u}_1 + (\mathbf{b} \cdot \mathbf{u}_2) \mathbf{u}_2 \\ &= (\mathbf{u}_1^\top \mathbf{b}) \mathbf{u}_1 + (\mathbf{u}_2^\top \mathbf{b}) \mathbf{u}_2 \\ &= \mathbf{u}_1 (\mathbf{u}_1^\top \mathbf{b}) + \mathbf{u}_2 (\mathbf{u}_2^\top \mathbf{b}) \\ &= (\mathbf{u}_1 \mathbf{u}_1^\top) \mathbf{b} + (\mathbf{u}_2 \mathbf{u}_2^\top) \mathbf{b} \\ &= (\mathbf{u}_1 \mathbf{u}_1^\top + \mathbf{u}_2 \mathbf{u}_2^\top) \mathbf{b} \\ &= [\mathbf{u}_1 \ \mathbf{u}_2] \begin{bmatrix} \mathbf{u}_1^\top \\ \mathbf{u}_2^\top \end{bmatrix} \mathbf{b} = \mathbf{U} \mathbf{U}^\top \mathbf{b} \Rightarrow \text{Linear Transformation!} \end{aligned} \quad (48)$$

2.19 Orthogonal Projection Perspective

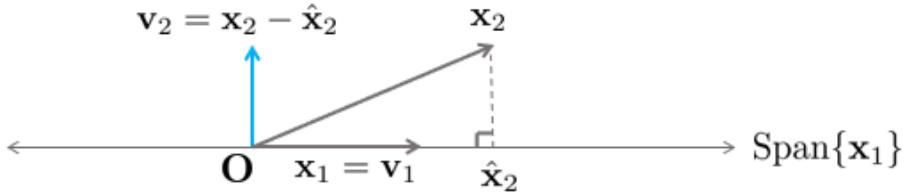
정규직교인 열벡터를 가지는 행렬 $\mathbf{A} = \mathbf{U} = [\mathbf{u}_1 \ \mathbf{u}_2]$ 가 있을 때 \mathbf{b} 벡터를 $\text{Col } \mathbf{A}$ 공간으로 정사영시키는 경우

$$\hat{\mathbf{b}} = \mathbf{A}\hat{\mathbf{x}} = \mathbf{A}(\mathbf{A}^\top \mathbf{A})^{-1}\mathbf{A}^\top \mathbf{b} = f(\mathbf{b}) \quad (49)$$

행렬 $\mathbf{C} = \mathbf{A}^\top \mathbf{A}$ 는 $\mathbf{C} = \begin{bmatrix} \mathbf{u}_1^\top \\ \mathbf{u}_2^\top \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 \end{bmatrix} = \mathbf{I}$ 와 같은 성질을 지니게 되고 따라서 다음과 같은 공식이 성립한다.

$$\hat{\mathbf{b}} = \mathbf{A}\hat{\mathbf{x}} = \mathbf{A}(\mathbf{A}^\top \mathbf{A})^{-1}\mathbf{A}^\top \mathbf{b} = \mathbf{A}(\mathbf{I})^{-1}\mathbf{A}^\top \mathbf{b} = \mathbf{A}\mathbf{A}^\top \mathbf{b} = \mathbf{U}\mathbf{U}^\top \mathbf{b} \quad (50)$$

2.20 Gram-Schmidt Orthogonalization



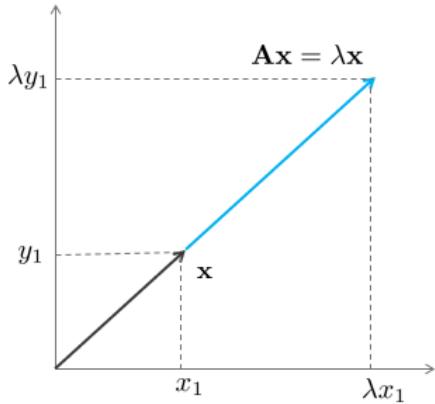
벡터 $\mathbf{x}_1 = \begin{bmatrix} 3 \\ 6 \\ 0 \end{bmatrix}$, $\mathbf{x}_2 = \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}$ 로 인해 $\text{Span}\{\mathbf{x}_1\}$ 부분공간 $Wx_1 = \text{Span}[\mathbf{x}_1 \ \mathbf{x}_2]$ 가 있을 때 두 벡터의 내적 $\mathbf{x}_1 \cdot \mathbf{x}_2 = 15 \neq 0$ 으로 두 벡터는 수직이 아니다.

이 때 벡터 $\mathbf{v}_1 = \mathbf{x}_1$ 이라고 하고 \mathbf{v}_2 를 \mathbf{x}_1 에 수직인 \mathbf{x}_2 의 성분이라고 했을 때

$$\mathbf{v}_2 = \mathbf{x}_2 - \frac{\mathbf{x}_2 \cdot \mathbf{x}_1}{\mathbf{x}_1 \cdot \mathbf{x}_1} \mathbf{x}_1 = \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix} - \frac{15}{45} \begin{bmatrix} 3 \\ 6 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} \quad (51)$$

가 된다. 이 때 벡터 $\mathbf{v}_1, \mathbf{v}_2$ 는 부분공간 W 의 직교기저벡터가 된다.

3 Eigenvectors and Eigenvalues



정방행렬 $\mathbf{A} \in \mathbb{R}^{n \times n}$ 에 대한 고유벡터(eigenvector)는 $\mathbf{Ax} = \lambda\mathbf{x}$ 를 만족하는 0이 아닌 벡터 $\mathbf{x} \in \mathbb{R}^n$ 을 말한다. 이 때 λ 는 행렬 \mathbf{A} 의 고유값(eigenvalue)이라고 한다.

$\mathbf{Ax} = \lambda\mathbf{x}$ 는 다음과 같이 다시 나타낼 수 있다.

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{x} = 0 \quad (52)$$

이 때, 위 시스템이 \mathbf{x} 가 0이 아닌 비자명해를 가지고 있는 경우에만 λ 값이 행렬 \mathbf{A} 에 대한 고유값이 된다. 위와 같은 동차 선형시스템이 비자명해를 가지기 위해서는 $\mathbf{A} - \lambda\mathbf{I}$ 가 선형의존(Linearly Dependent) 해야 무수히 많은 해를 가진다.

3.1 Null Space

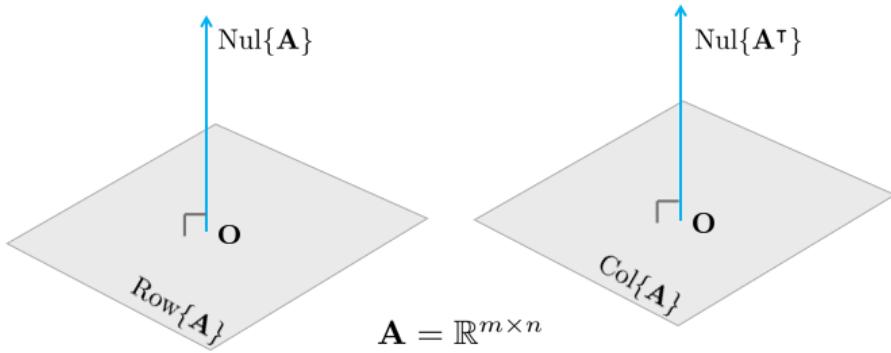
행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 의 동차 선형시스템(Homogeneous Linear System) $\mathbf{Ax} = 0$ 의 해 집합을 영공간(Null Space)라고 한다. $\text{Nul } \mathbf{A}$ 로 표기한다.

$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1^\top \\ \mathbf{a}_2^\top \\ \vdots \\ \mathbf{a}_m^\top \end{bmatrix}$ 일 때 벡터 \mathbf{x} 는 다음을 만족해야 한다.

$$\mathbf{a}_1^\top \mathbf{x} = \mathbf{a}_2^\top \mathbf{x} = \cdots = \mathbf{a}_m^\top \mathbf{x} = 0 \quad (53)$$

즉, \mathbf{x} 는 모든 \mathbf{A} 의 행벡터(Row Vector)과 직교해야 한다.

3.2 Orthogonal Complement



벡터 \mathbf{z} 가 부분공간 $W \in \mathbb{R}^n$ 의 모든 벡터와 직교하면 \mathbf{z} 는 부분공간 W 와 직교한다고 말할 수 있다. 부분공간 W 와 직교하는 모든 벡터 \mathbf{z} 의 집합을 직교여공간(Orthogonal Complement)라고 부르며 W^\perp 로 표시한다.

부분공간 W 의 직교여공간 W^\perp 에 위치한 벡터 $\mathbf{x} \in \mathbb{R}^n$ 은 부분공간 W 를 Span하는 모든 벡터들과 직교한다.

W^\perp is a subspace of \mathbb{R}^n .

$$\begin{aligned} \text{Nul } \mathbf{A} &= (\text{Row } \mathbf{A})^\perp \\ \text{Nul } \mathbf{A}^T &= (\text{Col } \mathbf{A})^\perp \end{aligned} \quad (54)$$

3.3 Characteristic Equation

방정식 $(\mathbf{A} - \lambda\mathbf{I})\mathbf{x} = 0$ 이 비자명해를 갖기 위해서는 $(\mathbf{A} - \lambda\mathbf{I})$ 행렬이 선형의존이어야 하고 이는 곧 역행렬이 존재하지 않아야 하는 것과 동치(Equivalent)이다. 만약 $(\mathbf{A} - \lambda\mathbf{I})$ 의 역행렬이 존재한다면 \mathbf{x} 는 자명해 이외에는 갖지 못한다.

$$\begin{aligned} (\mathbf{A} - \lambda\mathbf{I})^{-1}(\mathbf{A} - \lambda\mathbf{I})\mathbf{x} &= (\mathbf{A} - \lambda\mathbf{I})^{-1}0 \\ \mathbf{x} &= 0 \end{aligned} \quad (55)$$

따라서 행렬 \mathbf{A} 에 대하여 고유값과 고유벡터가 존재하기 위해서는 다음의 방정식이 항상 성립해야 한다.

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0 \quad (56)$$

위 방정식을 행렬 \mathbf{A} 의 특성방정식(Characteristic Equation)이라고 부른다.

3.4 Eigenspace

$(\mathbf{A} - \lambda \mathbf{x})\mathbf{x} = 0$ 에서 $(\mathbf{A} - \lambda \mathbf{x})$ 의 영공간(Null Space)를 고유값 λ 에 대한 고유공간(Eigenspace)라고 한다. λ 에 대한 고유공간의 차원이 1 이상인 경우, 고유공간 내에 있는 모든 벡터들에 대하여 다음이 성립한다.

$$T(\mathbf{x}) = \mathbf{Ax} = \lambda \mathbf{x} \quad (57)$$

3.5 Diagonalization

정방행렬 $\mathbf{A} \in \mathbb{R}^{n \times n}$ 이 주어졌고 $\mathbf{V} \in \mathbb{R}^{n \times n}$ 이고 $\mathbf{D} \in \mathbb{R}^{n \times n}$ 일 때

$$\mathbf{D} = \mathbf{V}^{-1} \mathbf{AV} \quad (58)$$

위와 같은 공식이 성립한다면 이를 정방행렬 \mathbf{A} 의 대각화(Diagonalization)라고 한다. 대각화는 모든 경우에 대해서 항상 가능한 것은 아니다. 행렬 \mathbf{A} 가 대각화되기 위해서는 역행렬이 존재하는 행렬 \mathbf{V} 가 존재해야 한다. 행렬 \mathbf{V} 가 역행렬이 존재하기 위해서는 \mathbf{V} 는 행렬 \mathbf{A} 와 같은 $\mathbb{R}^{n \times n}$ 크기의 정방행렬이어야 하고 n 개의 선형독립인 열벡터를 가지고 있어야 한다. 이 때, \mathbf{V} 의 각 열은 행렬 \mathbf{A} 의 고유벡터가 된다. 만약 행렬 \mathbf{V} 가 존재하는 경우 행렬 \mathbf{A} 는 대각화 가능(Diagonalizable)하다고 한다.

3.6 Finding \mathbf{V} and \mathbf{D}

대각화 공식은 다음과 같이 다시 작성할 수 있다.

$$\mathbf{D} = \mathbf{V}^{-1} \mathbf{AV} \Rightarrow \mathbf{VD} = \mathbf{AV} \quad (59)$$

$$\begin{aligned} \text{이 때, } \mathbf{V} &= [\mathbf{v}_1 \ \mathbf{v}_2 \ \cdots \ \mathbf{v}_n] \text{이고 } \mathbf{D} = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \lambda_n \end{bmatrix} \text{이라고 하면} \\ \mathbf{AV} &= \mathbf{A} [\mathbf{v}_1 \ \mathbf{v}_2 \ \cdots \ \mathbf{v}_n] = [\mathbf{Av}_1 \ \mathbf{Av}_2 \ \cdots \ \mathbf{Av}_n] \\ \mathbf{VD} &= [\mathbf{v}_1 \ \mathbf{v}_2 \ \cdots \ \mathbf{v}_n] \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \lambda_n \end{bmatrix} \\ &= [\lambda_1 \mathbf{v}_1 \ \lambda_2 \mathbf{v}_2 \ \cdots \ \lambda_n \mathbf{v}_n] \\ \mathbf{AV} &= \mathbf{VD} \Leftrightarrow [\mathbf{Av}_1 \ \mathbf{Av}_2 \ \cdots \ \mathbf{Av}_n] = [\lambda_1 \mathbf{v}_1 \ \lambda_2 \mathbf{v}_2 \ \cdots \ \lambda_n \mathbf{v}_n] \end{aligned} \quad (60)$$

위 공식과 같이

$$\mathbf{Av}_1 = \lambda_1 \mathbf{v}_1, \mathbf{Av}_2 = \lambda_2 \mathbf{v}_2, \dots, \mathbf{Av}_n = \lambda_n \mathbf{v}_n \quad (61)$$

각각의 열이 모두 동일해야 한다. 즉, 벡터 \mathbf{v}_i 는 행렬 \mathbf{A} 에 대한 고유벡터가 되어야 하고 스칼라 λ_i 는 행렬 \mathbf{A} 에 대한 고유값이 되어야 한다. 이에 따라 대각행렬 \mathbf{D} 는 고유값들을 대각성분으로 포함하고 있는 행렬이 된다. 결론적으로 정방행렬 $\mathbf{A} \in \mathbb{R}^{n \times n}$ 가 대각화 가능한가 안한가에 대한 질문은 n 개의 고유벡터가 존재하는가 안하는가에 대한 질문과 동치이다.

3.7 Eigendecomposition

정방행렬 \mathbf{A} 가 대각화 가능한 경우 $\mathbf{D} = \mathbf{V}^{-1} \mathbf{AV}$ 공식이 성립한다. 이 공식을 다시 작성하면 다음과 같다.

$$\mathbf{A} = \mathbf{VDV}^{-1} \quad (62)$$

이를 행렬 \mathbf{A} 에 대한 고유값 분해(Eigendecomposition)라고 한다. 행렬 \mathbf{A} 가 대각화 가능하다는 의미는 행렬 \mathbf{A} 가 고유값 분해 가능하다는 말과 동치이다.

3.8 Linear Transformation via Eigendecomposition

정방행렬 \mathbf{A} 가 대각화 가능한 경우 $\mathbf{A} = \mathbf{VDV}^{-1}$ 과 같이 고유값 분해가 가능하다. 이 때 선형 변환 $T(\mathbf{x}) = \mathbf{Ax}$ 을 생각해보면 다음과 같이 표현할 수 있다.

$$T(\mathbf{x}) = \mathbf{Ax} = \mathbf{VDV}^{-1}\mathbf{x} = \mathbf{V}(\mathbf{D}(\mathbf{V}^{-1}\mathbf{x})) \quad (63)$$

3.9 Change of Basis

예를 들어 $\mathbf{Av}_1 = -1\mathbf{v}_1$, $\mathbf{Av}_2 = 2\mathbf{v}_2$ 가 성립한다고 가정하고 $T(\mathbf{x}) = \mathbf{Ax} = \mathbf{VDV}^{-1}\mathbf{x} = \mathbf{V}(\mathbf{D}(\mathbf{V}^{-1}\mathbf{x}))$ 에서 $\mathbf{y} = \mathbf{V}^{-1}\mathbf{x}$ 라고 가정하면

$$\mathbf{Vy} = \mathbf{x} \quad (64)$$

의 관계가 성립한다. 이 때, 벡터 \mathbf{y} 는 벡터 \mathbf{x} 의 고유벡터 $\{\mathbf{v}_1, \mathbf{v}_2\}$ 에 대한 새로운 좌표를 의미한다.

$$\mathbf{x} = \begin{bmatrix} 4 \\ 3 \end{bmatrix} = 4 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 3 \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \end{bmatrix} = \mathbf{Vy} = [\mathbf{v}_1 \ \mathbf{v}_2] \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = 2\mathbf{v}_1 + 1\mathbf{v}_2 \Rightarrow \mathbf{y} = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \quad (65)$$

3.10 Element-wise Scaling

위 과정을 통해 \mathbf{y} 값을 구하고 나면 $T(\mathbf{x}) = \mathbf{V}(\mathbf{D}(\mathbf{V}^{-1}\mathbf{x}))$ 는 $T(\mathbf{x}) = \mathbf{V}(\mathbf{Dy})$ 로 표현할 수 있다. 이 때 $\mathbf{z} = \mathbf{Dy}$ 라고 하면 벡터 \mathbf{z} 는 단순히 벡터 \mathbf{y} 를 행렬의 대각 원소의 크기만큼 스케일드한 벡터가 된다.

3.11 Back to Original Basis

위 과정까지 진행했으면 $T(\mathbf{x}) = \mathbf{V}(\mathbf{Dy}) = \mathbf{Vz}$ 와 같이 나타낼 수 있고 이 때 벡터 \mathbf{z} 는 여전히 새로운 기저 벡터 $\{\mathbf{v}'_1, \mathbf{v}'_2\}$ 를 기반으로 하면 좌표가 된다. **\mathbf{Vz} 연산은 벡터 \mathbf{z} 를 다시 원래 기저벡터의 좌표로 변환하는 역할을 한다.** 벡터 \mathbf{Vz} 는 기존의 기저벡터 $\{\mathbf{v}_1, \mathbf{v}_2\}$ 의 선형결합이 된다.

$$\mathbf{Vz} = [\mathbf{v}_1 \ \mathbf{v}_2] \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \mathbf{v}_1 z_1 + \mathbf{v}_2 z_2 \quad (66)$$

지금까지의 과정을 고유값 분해를 통한 선형 변환이라고 한다.

3.12 Linear Transformation via \mathbf{A}^k

여러번의 변환이 중첩된 $\mathbf{A} \times \mathbf{A} \times \cdots \times \mathbf{A}x = \mathbf{A}^k\mathbf{x}$ 를 생각해보자. 이 때, 행렬 \mathbf{A} 가 대각화 가능하다면 \mathbf{A} 를 고유값 분해할 수 있고 이 때, \mathbf{A}^k 는 다음과 같이 분해할 수 있다.

$$\mathbf{A}^k = (\mathbf{VDV}^{-1})(\mathbf{VDV}^{-1}) \cdots (\mathbf{VDV}^{-1}) = \mathbf{VD}^k\mathbf{V}^{-1} \quad (67)$$

이 때 \mathbf{D}^k 는 다음과 같이 표현된다.

$$\mathbf{D}^k = \begin{bmatrix} \lambda_1^k & 0 & \cdots & 0 \\ 0 & \lambda_2^k & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \lambda_n^k \end{bmatrix} \quad (68)$$

3.13 Geometric Multiplicity and Algebraic Multiplicity

정방행렬 $\mathbf{A} \in \mathbb{R}^{n \times n}$ 이 있을 때 \mathbf{A} 가 대각화 가능한지 안한지 판단을 해야하는 경우 일반적으로 판별식을 사용하여 판단한다.

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0 \quad (69)$$

예를 들어 $n = 5$ 인 정방행렬 \mathbf{A} 가 있을 때, $\det(\mathbf{A} - \lambda\mathbf{I})$ 는 5차 다항식이 나오게 된다. 5차 다항식은 일반적으로 5개의 해를 가지고 있지만 실수만 고려하는 경우 5개의 해가 계산되지 않을 수 있다. 즉, **실근이 5개가 나오지 않는 경우 $n = 5$ 개의 선형독립인 고유벡터가 나오지 않으므로 대각화가 불가능하다.**

만약 실근 중 중근이 포함되는 경우, 예를 들어 $(\lambda - 2)^2(\lambda - 3) = 0$ 과 같이 $\lambda = 2$ 가 중근인 경우, $\lambda = 2$ 로 인해 생성되는 고유공간(Eigenspace)의 차원이 최대 $\lambda = 2$ 가 가지는 중근의 개수까지 가질 수 있다.

중근이 아닌 일반 실근의 경우 최대 1차원의 고유공간을 가질 수 있다. 즉, 중근이 포함된 경우 고유공간의 차원이 최대 $n = 5$ 까지 생성될 수 있는데 $n = 5$ 를 만족하지 못하는 경우에는 대각화가 불가능하다.

이와 같이 대수적으로 판별식을 인수분해했을 때, 중근이 생기는 경우 중근의 대수 중복도(Algebraic Multiplicity)와 이로 인해 Span되는 고유공간의 기하 중복도(Geometric Multiplicity)가 일치해야 n 개의 독립적인 고유벡터가 생성될 수 있고 행렬 \mathbf{A} 의 대각화가 가능하다.

4 Singular Value Decomposition

$$\mathbf{A} = \mathbf{UDV}^\top = (\mathbf{A} \in \mathbb{R}^{m \times n})$$

행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 이 주어졌을 때 특이값 분해(Singular Value Decomposition, SVD)는 다음과 같이 나타낼 수 있다.

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^\top \quad (70)$$

이 때, $\mathbf{U} \in \mathbb{R}^{m \times m}$, $\mathbf{V} \in \mathbb{R}^{n \times n}$ 인 행렬이며 이들은 각 열이 Col \mathbf{A} 와 Row \mathbf{A} 에 의 정규직교기저벡터(Orthonormal Basis)로 구성되어 있다. $\Sigma \in \mathbb{R}^{m \times n}$ 은 대각행렬이며 대각 성분들이 $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{\min(m,n)}$ 특이값이며 큰 값부터 내림차순으로 정렬된 행렬이다.

4.1 SVD as Sum of Outer Products

행렬 \mathbf{A} 는 다음과 같이 Outer Products의 합으로 표현할 수 있다.

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^\top = \sum_{i=1}^n \sigma_i \mathbf{u}_i \mathbf{v}_i^\top, \quad \text{where } \sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \quad (71)$$

이 때 위 식을 다시 행렬로 합성하면 $\mathbf{U} \in \mathbb{R}^{m \times m} \rightarrow \mathbf{U}' \in \mathbb{R}^{m \times n}$ 그리고 $\mathbf{D} \in \mathbb{R}^{m \times n} \rightarrow \mathbf{D}' \in \mathbb{R}^{n \times n}$ 과 같이 행렬 \mathbf{V}^\top 의 차원에 맞게 다시 합성할 수 있는데 이를 **Reduced Form of SVD**이라고 한다.

$$\mathbf{A} = \mathbf{U}'\mathbf{D}'\mathbf{V}^\top \quad (72)$$

4.2 Another Perspective of SVD

행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 에 대해 Gram-Schmidt Orthogonalization을 사용하면 Col \mathbf{A} 에 대한 정규직교기저벡터 $\mathbf{u}_1, \dots, \mathbf{u}_n$ 과 Row \mathbf{A} 에 대한 정규직교기저벡터 $\mathbf{v}_1, \dots, \mathbf{v}_n$ 을 구할 수 있다. 하지만 이렇게 계산한 정규직교기저벡터 $\mathbf{u}_i, \mathbf{v}_i$ 는 유일하지 않다.

Reduced Form of SVD를 사용하면 행렬 $\mathbf{U} = [\mathbf{u}_1 \ \mathbf{u}_2 \ \dots \ \mathbf{u}_n] \in \mathbb{R}^{m \times n}$ 과 $\mathbf{V} = [\mathbf{v}_1 \ \mathbf{v}_2 \ \dots \ \mathbf{v}_n] \in \mathbb{R}^{n \times n}$ 그리고 $\Sigma = \begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \sigma_n \end{bmatrix} \in \mathbb{R}^{n \times n}$ 일 때

$$\begin{aligned} \mathbf{AV} &= \mathbf{A} [\mathbf{v}_1 \quad \mathbf{v}_2 \quad \cdots \quad \mathbf{v}_n] = [\mathbf{Av}_1 \quad \mathbf{Av}_2 \quad \cdots \quad \mathbf{Av}_n] \\ \mathbf{U}\Sigma &= [\mathbf{u}_1 \quad \mathbf{u}_2 \quad \cdots \quad \mathbf{u}_n] \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_n \end{bmatrix} \\ &\quad [\sigma_1\mathbf{u}_1 \quad \sigma_2\mathbf{u}_2 \quad \cdots \quad \sigma_n\mathbf{u}_n] \\ \mathbf{AV} &= \mathbf{U}\Sigma \Leftrightarrow [\mathbf{Av}_1 \quad \mathbf{Av}_2 \quad \cdots \quad \mathbf{Av}_n] = [\sigma_1\mathbf{u}_1 \quad \sigma_2\mathbf{u}_2 \quad \cdots \quad \sigma_n\mathbf{u}_n] \end{aligned} \tag{73}$$

위 식을 간결하게 나타내면 다음과 같다.

$$\mathbf{AV} = \mathbf{U}\Sigma \Leftrightarrow \mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^\top \tag{74}$$

4.3 Computing SVD

행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 에 대하여 \mathbf{AA}^\top 와 $\mathbf{A}^\top\mathbf{A}$ 는 다음과 같이 고유값분해할 수 있다.

$$\begin{aligned} \mathbf{AA}^\top &= \mathbf{U}\Sigma\mathbf{V}^\top\mathbf{V}\Sigma^\top\mathbf{U}^\top = \mathbf{U}\Sigma\Sigma^\top\mathbf{U}^\top = \mathbf{U}\Sigma^2\mathbf{U}^\top \\ \mathbf{A}^\top\mathbf{A} &= \mathbf{V}\Sigma^\top\mathbf{U}^\top\mathbf{U}\Sigma\mathbf{V}^\top = \mathbf{V}\Sigma^\top\Sigma\mathbf{U}^\top = \mathbf{V}\Sigma^2\mathbf{V}^\top \end{aligned} \tag{75}$$

이 때 계산되는 행렬 \mathbf{U}, \mathbf{V} 은 직교하는 고유벡터를 각 열의 성분으로 하는 행렬이며 대각행렬 Σ^2 의 각 성분은 항상 0보다 큰 양수의 값을 가진다. 그리고 \mathbf{AA}^\top 와 $\mathbf{A}^\top\mathbf{A}$ 를 통해 계산되는 Σ^2 의 값은 동일하다.

4.4 Diagonalization of Symmetric Matrices

일반적으로 정방행렬 $\mathbf{A} \in \mathbb{R}^{n \times n}$ 이 n 개의 선형독립인 고유벡터를 가지고 있을 경우 대각화 가능하다. 그리고 대칭행렬 $\mathbf{S} \in \mathbb{R}^{n \times n}, \mathbf{S}^\top = \mathbf{S}$ 는 항상 대각화 가능하다. 추가적으로 대칭행렬 \mathbf{S} 의 고유벡터는 항상 서로에게 직교하므로 직교대각화(Orthogonally Diagonalizable)가 가능하다.

4.5 Spectral Theorem of Symmetric Matrices

$\mathbf{S}^\top = \mathbf{S}$ 를 만족하는 대칭행렬 \mathbf{S} 가 주어졌을 때 \mathbf{S} 는 n 개의 중근을 포함한 실수의 고유값이 존재한다. 또한, 고유공간의 차원은 기하 중복도(Algebraic Multiplicity)와 기하 중복도(Geometric Multiplicity)와 같아야 한다. 서로 다른 λ 값 들에 대한 고유공간들은 서로 직교한다. 결론적으로 대칭행렬 \mathbf{S} 은 직교대각화가 가능하다.

4.6 Spectral Decomposition

대칭행렬 \mathbf{S} 의 고유값 분해는 Spectral Decomposition이라고 불린다. 이는 다음과 같이 나타낼 수 있다.

$$\begin{aligned} \mathbf{S} &= \mathbf{UDU}^{-1} = \mathbf{UDU}^\top = [\mathbf{u}_1 \quad \mathbf{u}_2 \quad \cdots \quad \mathbf{u}_n] \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \lambda_n \end{bmatrix} \begin{bmatrix} \mathbf{u}_1^\top \\ \mathbf{u}_2^\top \\ \vdots \\ \mathbf{u}_n^\top \end{bmatrix} \\ &= [\lambda_1\mathbf{u}_1 \quad \lambda_2\mathbf{u}_2 \quad \cdots \quad \lambda_n\mathbf{u}_n] \begin{bmatrix} \mathbf{u}_1^\top \\ \mathbf{u}_2^\top \\ \vdots \\ \mathbf{u}_n^\top \end{bmatrix} \\ &= \lambda_1\mathbf{u}_1\mathbf{u}_1^\top + \lambda_2\mathbf{u}_2\mathbf{u}_2^\top + \cdots + \lambda_n\mathbf{u}_n\mathbf{u}_n^\top \end{aligned} \tag{76}$$

위 식에서 각 항 $\lambda_i\mathbf{u}_j\mathbf{u}_j^\top$ 은 \mathbf{u}_j 에 의해 Span된 부분공간에 프로젝션된 다음 고유값 λ_i 만큼 스케일된 벡터로 볼 수 있다.

4.7 Positive (Semi-)Definite Matrices

정방행렬 $\mathbf{A} \in \mathbb{R}^{n \times n}$ 이 있을 때 0이 아닌 모든 벡터 $\forall \mathbf{x} \neq 0$ 에 대하여 $\mathbf{x}^\top \mathbf{A} \mathbf{x} > 0$ 을 만족하는 경우 \mathbf{A} 를 **Positive Definite** 행렬이라고 한다. 만약 $\mathbf{x}^\top \mathbf{A} \mathbf{x} \geq 0$ 인 경우 **Positive Semi-Definite** 행렬이라고 한다. 정방행렬 \mathbf{A} 가 Postivie Definite인 경우 \mathbf{A} 의 고유값은 항상 모두 양수이다.

4.8 Symmetric Positive Definite Matrices

행렬 $\mathbf{S} \in \mathbb{R}^{n \times n}$ 이 대칭이면서 Positive Definite인 경우 Spectral Decomposition의 모든 고유값은 항상 양수가 된다.

$$\begin{aligned} \mathbf{S} = \mathbf{U} \mathbf{D} \mathbf{U}^{-1} &= \mathbf{U} \mathbf{D} \mathbf{U}^\top = [\mathbf{u}_1 \quad \mathbf{u}_2 \quad \cdots \quad \mathbf{u}_n] \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \lambda_n \end{bmatrix} [\mathbf{u}_1^\top \quad \mathbf{u}_2^\top \quad \cdots \quad \mathbf{u}_n^\top] \\ &= \lambda_1 \mathbf{u}_1 \mathbf{u}_1^\top + \lambda_2 \mathbf{u}_2 \mathbf{u}_2^\top + \cdots + \lambda_n \mathbf{u}_n \mathbf{u}_n^\top \\ \text{where, } \lambda_j &> 0, \forall j = 1, \dots, n \end{aligned} \quad (77)$$

4.9 Back to Computing SVD

행렬 \mathbf{A} 에 대하여 $\mathbf{A} \mathbf{A}^\top = \mathbf{A}^\top \mathbf{A} = \mathbf{S}$ 인 대칭행렬이 존재할 때 \mathbf{S} 가 Positive (Semi-)Definite한 경우

$$\begin{aligned} \mathbf{x}^\top \mathbf{A} \mathbf{A}^\top \mathbf{x} &= (\mathbf{A}^\top \mathbf{x})^\top (\mathbf{A}^\top \mathbf{x}) = \|\mathbf{A}^\top \mathbf{x}\| \geq 0 \\ \mathbf{x}^\top \mathbf{A}^\top \mathbf{A} \mathbf{x} &= (\mathbf{A} \mathbf{x})^\top (\mathbf{A} \mathbf{x}) = \|\mathbf{A} \mathbf{x}\|^2 \geq 0 \end{aligned} \quad (78)$$

즉, $\mathbf{A} \mathbf{A}^\top = \mathbf{U} \Sigma^2 \mathbf{U}^\top$ 와 $\mathbf{A}^\top \mathbf{A} = \mathbf{V} \Sigma^2 \mathbf{V}^\top$ 에서 Σ^2 의 값은 항상 양수가 된다.

임의의 직각 행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 에 대하여 특이값 분해는 언제나 존재한다. $\mathbf{A} \in \mathbb{R}^{n \times n}$ 행렬의 경우 고유값 분해가 존재하지 않을 수 있지만 특이값 분해는 항상 존재한다. 대칭이면서 동시에 Positive Definite인 정방 행렬 $\mathbf{S} \in \mathbb{R}^{n \times n}$ 은 항상 고유값 분해값이 존재하며 이는 특이값 분해와 동일하다.

4.10 Eigendecomposition in Machine Learning

일반적으로 머신러닝에서는 대칭이고 Positive Definite인 행렬을 다룬다. 예를 들면, $\mathbf{A} \in \mathbb{R}^{10 \times 3}$ 인 행렬이 있고 각 열은 사람을 의미하고 각 행은 Feature를 의미한다고 가정했을 때, $\mathbf{A}^\top \mathbf{A} \in \mathbb{R}^{3 \times 3}$ 는 각 사람들 간 유사도를 의미하고 $\mathbf{A} \mathbf{A}^\top \in \mathbb{R}^{10 \times 10}$ 는 각 Feature들의 상관관계를 의미한다. 이 때, $\mathbf{A} \mathbf{A}^\top$ 는 주성분분석 (Principal Component Analysis)에서 Covariance Matrix를 구할 때 사용된다.

4.11 Low Rank Approximation of a Matrix

행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 이 주어졌을 때 예를 들어, \mathbf{A} 의 원래 rank가 r 일 때, 행렬 \mathbf{A} 에서 rank를 r 이하를 가진 근사행렬 $\hat{\mathbf{A}}$ 을 찾는 Low Rank Approximation을 수행할 수 있다.

$$\begin{aligned} \hat{\mathbf{A}}_r &= \arg \min_{\hat{\mathbf{A}}_r} \|\mathbf{A} - \hat{\mathbf{A}}_r\|_F, \text{ subject to } \text{rank} \mathbf{A}_r \leq r \\ \hat{\mathbf{A}}_r &= \sum_{i=1}^r \sigma_i \mathbf{u}_i \mathbf{v}_i^\top \quad \text{where, } \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \end{aligned} \quad (79)$$

4.12 Dimension Reducing Transformation

Feature-by-data item 행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 이 주어졌을 때 $\mathbf{G} \in \mathbb{R}^{m \times r}, r < m$ 인 변환 $\mathbf{G}^\top : \mathbf{x} \in \mathbb{R}^m \mapsto \mathbf{y} \in \mathbb{R}^r$ 을 생각해보면

$$\mathbf{y}_i = \mathbf{G}^\top \mathbf{a}_i \quad (80)$$

가 성립하고 \mathbf{G} 의 각 열들은 정규직교벡터이며 데이터의 유사도 행렬 $\mathbf{S} = \mathbf{A}^\top \mathbf{A}$ 의 유사도를 보존하는 변환 \mathbf{G} 를 차원 축소 변환(Dimension-Reducing Transformation)이라고 한다.

$$\begin{aligned} \mathbf{Y} &= \mathbf{G}^\top \mathbf{A} \\ \mathbf{Y}^\top \mathbf{Y} &= (\mathbf{G}^\top \mathbf{A})^\top \mathbf{G}^\top \mathbf{A} = \mathbf{A}^\top \mathbf{G} \mathbf{G}^\top \mathbf{A} \end{aligned} \quad (81)$$

이 때 차원축소변환 $\hat{\mathbf{G}}$ 은 다음과 같이 추정할 수 있다.

$$\hat{\mathbf{G}} = \arg \min_{\mathbf{G}} \|S - \mathbf{A}^T \mathbf{G} \mathbf{G}^T \mathbf{A}\|_F \text{ subject to } \mathbf{G}^T \mathbf{G} = \mathbf{I}_k \quad (82)$$

주어진 행렬 $\mathbf{A} = \mathbf{U} \Sigma \mathbf{V}^T = \sum_{i=1}^n \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ 에 대하여 최적의 해는 다음과 같다.

$$\hat{\mathbf{G}} = \mathbf{U}_r = [\mathbf{u}_1 \quad \mathbf{u}_2 \quad \cdots \quad \mathbf{u}_r] \quad (83)$$

5 Derivative of multi-variable function

5.1 Gradient

임의의 벡터 $\mathbf{x} \in \mathbb{R}^n$ 에 대하여 $f(\mathbf{x}) \in \mathbb{R}$ 를 만족하는 다변수 스칼라 함수 $f(\mathbf{x})$ 가 주어졌다고 하자.

$$f : \mathbb{R}^n \mapsto \mathbb{R} \quad (84)$$

$f(\mathbf{x})$ 에 대한 1차 편미분은 벡터가 되고 이는 그레디언트(gradient)라고 불린다.

$$\boxed{\nabla \mathbf{f} = \begin{pmatrix} \frac{\partial f}{\partial x_1} & \cdots & \frac{\partial f}{\partial x_n} \end{pmatrix} \in \mathbb{R}^{1 \times n}} \quad (85)$$

5.2 Jacobian matrix

임의의 벡터 $\mathbf{x} \in \mathbb{R}^n$ 에 대하여 $f(\mathbf{x}) \in \mathbb{R}^m$ 를 만족하는 다변수 벡터 함수 $f(\mathbf{x})$ 가 주어졌다고 하자.

$$f : \mathbb{R}^n \mapsto \mathbb{R}^m \quad (86)$$

이 때, $f(\cdot)$ 의 1차 편미분은 행렬이 되고 이를 특별히 자코비안(jacobian) 행렬이라고 한다.

$$\boxed{\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{m \times n}} \quad (87)$$

이를 통해 자코비안 행렬의 각 행벡터(row vector)는 함수 $f_m(\cdot)$ 에 대한 그레디언트라는 것을 알 수 있다. 위 식을 미소변화량 \mathbf{h} 를 사용하여 나타내면 다음과 같다.

$$\mathbf{J} = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \triangleq \lim_{\mathbf{h} \rightarrow \mathbf{0}} \frac{f(\mathbf{x} + \mathbf{h}) - f(\mathbf{x})}{\mathbf{h}} \in \mathbb{R}^{m \times n} \quad (88)$$

SLAM에서 자코비안은 에러 $\mathbf{e}(\mathbf{x})$ 를 최적화할 때 사용된다. SLAM에서 최적화하고자 하는 에러는 일반적으로 비선형 함수로 구성되어 있으며 크기가 작기 때문에 에러의 변화량 $\mathbf{e}(\mathbf{x} + \Delta \mathbf{x})$ 를 그대로 사용하지 않고 테일러 전개하여 근사식 $\mathbf{e}(\mathbf{x}) + \mathbf{J} \Delta \mathbf{x}$ 으로 표현하게 되는데 이 때 에러에 대한 자코비안 \mathbf{J} 가 유도된다. 그리고 근사식을 바탕으로 유도한 에러의 최적 증분량 $\Delta \mathbf{x}^* = (\mathbf{J}^T \mathbf{J})^{-1} \mathbf{J}^T \mathbf{b}$ 이 자코비안을 통해 구해지기 때문에 SLAM에서는 자코비안이 필수적으로 사용된다. 자세한 내용은 [SLAM] Errors and Jacobian Derivations for SLAM 정리 포스트를 참조하면 된다.

5.2.1 Toy example 1

만약 $\mathbf{x} = \{a, b, c\}$ 일 때 $f(\mathbf{x}) = f(a, b, c)$ 를 각각의 변수 a, b, c 에 대해 편미분하면 다음과 같다.

$$\mathbf{J} = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} = [\mathbf{J}_a \quad \mathbf{J}_b \quad \mathbf{J}_c] \quad (89)$$

$$\begin{aligned} \mathbf{J}_a &= \frac{\partial f(a, b, c)}{\partial a} \\ \mathbf{J}_b &= \frac{\partial f(a, b, c)}{\partial b} \\ \mathbf{J}_c &= \frac{\partial f(a, b, c)}{\partial c} \end{aligned} \quad (90)$$

만약 $a = a_0$ 로 계산값(=operating point)이 정해진 경우 자코비안은 다음과 같다.

$$\begin{aligned}\mathbf{J}_a &= \left. \frac{\partial f(a, b, c)}{\partial a} \right|_{a=a_0} \\ \mathbf{J}_b &= \left. \frac{\partial f(a, b, c)}{\partial b} \right|_{a=a_0, b=b_0} \\ \mathbf{J}_c &= \left. \frac{\partial f(a, b, c)}{\partial c} \right|_{a=a_0, c=c_0}\end{aligned}\tag{91}$$

위 첫번째 식은 $f(a, b, c)$ 를 a 에 대해 편미분한 후 $a = a_0$ 를 넣어 값을 계산하라는 의미이고 두번째와 세번째 식은 $a = a_0$ 로 값을 고정한 상태에서 각각 $b = b_0, c = c_0$ 에 대한 편미분을 수행하라는 의미이다.

5.2.2 Toy example 2

예를 들어 다음과 같은 3개의 연립 방정식이 주어졌다고 하자.

$$f(\mathbf{x}) = \begin{cases} f_1(\mathbf{x}) = ax^2 + 2bx + cy \\ f_2(\mathbf{x}) = dx^3 + ex \\ f_3(\mathbf{x}) = fx + gy^2 + hy \end{cases}\tag{92}$$

$\mathbf{x} = (x, y)$ 를 의미한다. 위 함수는 다음과 같이 쓸 수 있다.

$$f : \mathbb{R}^2 \mapsto \mathbb{R}^3\tag{93}$$

자코비안의 정의에 따라 이를 아래와 같이 쓸 수 있다.

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} \\ \frac{\partial f_3}{\partial x_1} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_3} \end{bmatrix} = \begin{bmatrix} 2ax + 2b & c \\ 3dx^2 + e & 0 \\ f & 2gy + h \end{bmatrix} \in \mathbb{R}^{3 \times 2}\tag{94}$$

5.3 Hessian matrix

임의의 벡터 $\mathbf{x} \in \mathbb{R}^n$ 에 대하여 $f(\mathbf{x}) \in \mathbb{R}$ 를 만족하는 다변수 스칼라 함수 $f(\mathbf{x})$ 가 주어졌다고 하자.

$$f : \mathbb{R}^n \mapsto \mathbb{R}\tag{95}$$

이 때, $f(\cdot)$ 의 2차 편미분은 행렬이 되고 이를 특별히 헤시안(hessian) 행렬이라고 한다. 헤시안 행렬은 일반적으로 대칭행렬의 형태를 띠고 있으며 다변수 벡터 함수가 아닌 다변수 스칼라 함수에 대한 2차 미분임에 유의한다.

$$\mathbf{H} = \boxed{\begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}} \in \mathbb{R}^{n \times n}\tag{96}$$

5.4 Laplacian

임의의 벡터 $\mathbf{x} \in \mathbb{R}^n$ 에 대하여 $f(\mathbf{x}) \in \mathbb{R}$ 를 만족하는 다변수 스칼라 함수 $f(\mathbf{x})$ 가 주어졌다고 하자.

$$f : \mathbb{R}^n \mapsto \mathbb{R}\tag{97}$$

$f(\mathbf{x})$ 에 대한 라플라시안(laplacian)은 각 입력 벡터에 따른 2차 편미분의 합으로 정의된다.

$$\nabla^2 \mathbf{f} = \frac{\partial^2 f}{\partial x_1^2} + \frac{\partial^2 f}{\partial x_2^2} + \cdots + \frac{\partial^2 f}{\partial x_n^2} \in \mathbb{R}^{1 \times n}\tag{98}$$

5.5 Taylor expansion

테일러 전개(expansion)은 미지의 함수 $f(x)$ 를 $x = a$ 지점에서 근사 다항함수로 표현하는 방법을 말한다. 이는 테일러 급수(series) 또는 테일러 근사(approximation)이라고도 불린다. $f(\cdot)$ 을 $x = a$ 부근에서 테일러 전개를 수행하면 다음과 같이 나타낼 수 있다.

$$f(x)|_{x=a} = f(a) + f'(a)(x - a) + \frac{1}{2!}f''(a)(x - a)^2 + \frac{1}{3!}f'''(a)(x - a)^3 + \dots \quad (99)$$

함수 $f(\cdot)$ 가 다변수 스칼라 함수일 경우 $\mathbf{x} = \mathbf{a}$ 지점에서 테일러 전개는 다음과 같이 쓸 수 있다.

$$f(\mathbf{x})|_{\mathbf{x}=\mathbf{a}} = f(\mathbf{a}) + \nabla f(\mathbf{x} - \mathbf{a}) + \frac{1}{2!}(\mathbf{x} - \mathbf{a})^\top \mathbf{H}(\mathbf{x} - \mathbf{a}) + \dots \quad (100)$$

이 때, ∇f 는 함수 $f(\cdot)$ 의 그레디언트(gradient) 의미하며 \mathbf{H} 는 해시안(hessian) 행렬을 의미한다.

6 Matrix Algebra

6.1 Identity Matrix

항등행렬(Identity Matrix)은 대각성분이 전부 1이고 나머지 성분이 전부 0인 $n \times n$ 크기의 정방행렬을 의미한다. 일반적으로 $\mathbf{I} \in \mathbb{R}^{n \times n}$ 으로 표현한다.

$$\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3 \times 3} \quad (101)$$

항등행렬에 임의의 벡터 $\mathbf{x} \in \mathbb{R}^n$ 을 곱하면 자기 자신이 도출된다.

$$\forall \mathbf{x} \in \mathbb{R}^n, \quad \mathbf{Ix} = \mathbf{x} \quad (102)$$

6.2 Transpose of Matrix

임의의 $m \times n$ 크기의 행렬 \mathbf{A} 가 주어졌을 때 \mathbf{A} 의 전치행렬(transpose matrix)은 \mathbf{A}^\top 과 같이 나타내고 이는 행과 열의 성분을 서로 바꾼 행렬을 의미한다.

$$[\mathbf{A}]_{ij} = a_{ij} \quad (103)$$

위와 같은 행렬에 대하여 \mathbf{A}^\top 는 다음과 같다.

$$[\mathbf{A}^\top]_{ij} = a_{ji} \quad (104)$$

즉, $\mathbf{A} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}$ 인 행렬에 대한 전치행렬은 $\mathbf{A}^\top = \begin{bmatrix} a & d \\ b & e \\ c & f \end{bmatrix}$ 가 된다.

6.3 Determinant of Matrix

행렬식(determinant)은 임의의 정방행렬 $\mathbf{A} \in \mathbb{R}^{n \times n}$ 을 하나의 스칼라 값에 대응시키는 함수를 의미한다. 스칼라 값의 크기 및 부호에 따라 해가 존재하는지 유무가 결정되며 행렬식이 0인 경우 해당 정방행렬은 역행렬이 존재하지 않는다. 행렬식은 일반적으로 $\det(\mathbf{A})$ 라고 표기하며 다음과 같다.

$$\det(\mathbf{A}) = \sum_{j=1}^n a_{ij} C_{ij} \quad (105)$$

- $C_{ij} = (-1)^{i+j} M_{ij}$

M_{ij} 는 \mathbf{A} 에서 i 행과 j 열을 제거한 부분 행렬(submatrix)에 대한 행렬식(determinant)을 의미하며 a_{ij} 에 대한 minor라고도 부른다. 그리고 C_{ij} 는 cofactor라고도 부른다. 자세한 내용은 [[4]]를 참고하면 된다.

2×2 크기의 정방행렬 $\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ 가 있을 때 $\det(\mathbf{A})$ 는 다음과 같다.

$$\det(\mathbf{A}) = \frac{1}{ad - bc} \quad (106)$$

임의의 정방행렬 $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$ 에 대하여 행렬식은 다음과 같은 성질을 지닌다.

$$\begin{aligned} \det(\mathbf{A}^\top) &= \det(\mathbf{A}) \\ \det(c\mathbf{A}) &= c^n \det(\mathbf{A}) \\ \det(\mathbf{AB}) &= \det(\mathbf{A})\det(\mathbf{B}) \\ \det(\mathbf{A}^{-1}) &= \frac{1}{\det(\mathbf{A})} \end{aligned} \quad (107)$$

6.4 Inverse Matrix

정방행렬 $\mathbf{A} \in \mathbb{R}^{n \times n}$ 에 대한 역행렬(Inverse Matrix) \mathbf{A}^{-1} 는 다음과 같이 정의된다.

$$\mathbf{A}^{-1}\mathbf{A} = \mathbf{AA}^{-1} = \mathbf{I} \quad (108)$$

2×2 크기의 정방행렬 $\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ 가 있을 때, 역행렬은 다음과 같이 정의된다.

$$\mathbf{A}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} \quad (109)$$

3×3 크기 이상의 정방행렬도 역행렬을 구할 수 있다. 역행렬은 정방행렬이면서 Full Rank인 행렬(= non-singular, $\det \mathbf{A} \neq 0$)에만 존재하며 역행렬이 존재하지 않는 행렬 \mathbf{A} 는 singular하다고 한다.

6.5 Trace of Matrix

Trace란 임의의 행렬 \mathbf{A} 가 주어졌을 때 행렬의 trace는 행렬의 대각 성분의 합을 의미하며 $\text{tr}(\mathbf{A})$ 와 같이 표기한다.

$$\text{tr}(\mathbf{A}) = \sum_i [\mathbf{A}]_{ii} \quad (110)$$

- $[\mathbf{A}]_{ij}$: 행렬 \mathbf{A} 의 i 행 j 열의 원소

Trace는 다음과 같은 성질을 지닌다.

$$\begin{aligned} \text{tr}(\mathbf{A}) &= \text{tr}(\mathbf{A}^\top) \\ \text{tr}(\mathbf{AB}) &= \text{tr}(\mathbf{BA}) \\ \text{tr}(\mathbf{A} + \mathbf{B}) &= \text{tr}(\mathbf{A}) + \text{tr}(\mathbf{B}) \\ \text{tr}(\mathbf{ABC}) &= \text{tr}(\mathbf{BCA}) = \text{tr}(\mathbf{CAB}) \\ \text{tr}(\mathbf{A}^\top \mathbf{B}) &= \sum_{i=1}^n \sum_{j=1}^n [\mathbf{A}]_{ij} [\mathbf{B}]_{ij} \\ \mathbf{a}^\top \mathbf{b} &= \text{tr}(\mathbf{ba}^\top) \end{aligned} \quad (111)$$

- \mathbf{a} : 임의의 벡터

6.6 Diagonal Matrix

$\mathbf{A} \in \mathbb{R}^{n \times n}$ 크기의 대각 행렬(Diagonal Matrix)은 대각 성분을 제외한 나머지 성분이 0인 행렬을 의미한다 ($a_{ij} = 0$ for $i \neq j$).

$$\mathbf{A} = \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{bmatrix} \quad (112)$$

대각 행렬의 역함수는 단순히 각 원소의 역수가 되기 때문에 매우 간단하게 역행렬을 구할 수 있다는 특징이 있다.

$$\mathbf{A}^{-1} = \begin{bmatrix} a_{11}^{-1} & 0 & \cdots & 0 \\ 0 & a_{22}^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn}^{-1} \end{bmatrix} \quad (113)$$

각각의 원소가 block matrix인 경우에도 동일하게 적용된다.

$$\mathbf{A}^{-1} = \begin{bmatrix} \mathbf{A}_{11}^{-1} & 0 & \cdots & 0 \\ 0 & \mathbf{A}_{22}^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{A}_{nn}^{-1} \end{bmatrix} \quad (114)$$

- A_{ii} : 대각 행렬을 만족하는 부분 행렬

이 때, 대각 행렬의 행렬식은 다음과 같다.

$$\det(\mathbf{A}) = \prod_{i=1}^n \det(\mathbf{A}_{ii}) \quad (115)$$

6.7 Idempotent Matrix

멱동 행렬(Idempotent Matrix)은 $n \times n$ 크기의 정방행렬이면서 다음을 만족하는 행렬을 의미한다.

$$\mathbf{A}^2 = \mathbf{A} \quad (116)$$

이는 $l \geq 1$ 에 대하여 $\mathbf{A}^l = \mathbf{A}$ 임을 의미한다. 최소제곱법에서 유도되는 프로젝션 행렬이 멱동 행렬에 해당한다.

$$\mathbf{A} = \mathbf{H}(\mathbf{H}^\top \mathbf{H})^{-1} \mathbf{H}^\top \quad (117)$$

7 Matrix Decompositions

7.1 LU decomposition

LU 분해는 $\mathbf{Ax} = \mathbf{b}$ 의 시스템에서 행렬 \mathbf{A} 를 하삼각(lower-triangle) 행렬 \mathbf{L} 과 상삼각(upper-triangle) 행렬 \mathbf{U} 의 곱으로 분해하는 방법이다.

$$\mathbf{A} = \mathbf{LU} = \mathbf{b} \quad (118)$$

LU 분해를 사용하면 다음과 같이 방정식이 변형된다.

$$\mathbf{Ax} = (\mathbf{LU})\mathbf{x} \quad (119)$$

이를 사용하여 [1] $\mathbf{Ly} = \mathbf{b}$ 방정식을 먼저 푼 후, [2] $\mathbf{Ux} = \mathbf{y}$ 를 순차적으로 계산할 수 있다. tridiagonal, band-diagonal 시스템에 효과적으로 사용할 수 있다.

$$\begin{aligned} \mathbf{L}(\mathbf{Ux}) &= \mathbf{b} \\ \mathbf{Ly} &= \mathbf{b} \quad \cdots [1] \\ \mathbf{Ux} &= \mathbf{y} \quad \cdots [2] \end{aligned} \quad (120)$$

위와 같이 행렬 \mathbf{A} 를 \mathbf{L}, \mathbf{U} 로 분해하면 \mathbf{x} 를 두 스텝에 걸쳐 구해야하지만 삼각행렬의 특성 상 \mathbf{x} 를 구하는 것이 훨씬 더 간단해진다.

7.1.1 PLU decomposition

만약 \mathbf{A} 가 아래와 같은 3×3 행렬이라고 가정해보자.

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1^\top \\ \mathbf{a}_2^\top \\ \mathbf{a}_3^\top \end{bmatrix} = \begin{bmatrix} 0 & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \quad (121)$$

LU 분해는 가우스 조던 소거법으로 \mathbf{L} 을 구하기 때문에 만약 \mathbf{A} 의 첫 번째 원소가 0으로 시작하는 경우 정상적으로 분해할 수 없다. 따라서 첫번째 행과 두번째 행의 순서를 변환하는 permutation 행렬 \mathbf{P} 를 앞에 곱해줘야 LU 분해를 수행할 수 있다.

$$\mathbf{PA} = \begin{bmatrix} & 1 \\ 1 & \end{bmatrix} \mathbf{A} = \begin{bmatrix} \mathbf{a}_2^\top \\ \mathbf{a}_1^\top \\ \mathbf{a}_3^\top \end{bmatrix} = \begin{bmatrix} * & * & * \\ 0 & * & * \\ * & * & * \end{bmatrix} \quad (122)$$

permutation 행렬은 직교행렬이고 직교행렬의 특성 상 $\mathbf{P} = \mathbf{P}^\top = \mathbf{P}^{-1}$ 이므로 이를 넘긴 후 전개하면 다음과 같다.

$$\mathbf{A} = \mathbf{PLU} \quad (123)$$

이와 같이 행벡터의 순서를 변경하는 \mathbf{P} 를 곱한 후 LU 분해를 수행하는 방법을 PLU 분해라고 한다.

7.1.2 LDU decomposition

LU 분해에서 \mathbf{L}, \mathbf{D} 행렬의 대각 성분을 1로 만들기 위해 중앙에 대각행렬 \mathbf{D} 를 별도로 분해하는 방법을 LDU 분해라고 한다. 따라서 모든 LU 행렬은 LDU 행렬로 분해할 수 있다.

$$\mathbf{A} = \mathbf{LU} = \mathbf{L}'\mathbf{DU}' \quad (124)$$

7.2 Cholesky decomposition

Cholesky 분해는 $\mathbf{Ax} = \mathbf{b}$ 시스템에서 **A가 대칭행렬이면서 동시에 positive(-semi) definite인 경우에 이를 하삼각(lower-triangle) 행렬 L의 곱으로 분해하는 방법을 말한다.**

$$\mathbf{A} = \mathbf{LL}^\top \quad (125)$$

Cholesky 분해는 수치적으로 안정하다는 특징이 있다.

7.2.1 Detailed explanation

임의의 3x3 대칭행렬 \mathbf{A} 가 주어졌다고 하자. 이를 cholesky 분해하면 다음과 같다.

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{21} & a_{22} & a_{32} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \mathbf{LL}^\top = \begin{bmatrix} l_{11} & & \\ l_{21} & l_{22} & \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} & l_{31} \\ & l_{22} & \\ & & l_{33} \end{bmatrix} \quad (126)$$

\mathbf{LL}^\top 을 자세히 전개하면 다음과 같다.

$$\begin{bmatrix} l_{11} & & \\ l_{21} & l_{22} & \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} & l_{31} \\ & l_{22} & l_{32} \\ & & l_{33} \end{bmatrix} = \begin{bmatrix} l_{11}^2 & & \\ l_{21}l_{11} & l_{21}^2 + l_{22}^2 & \\ l_{31}l_{11} & l_{31}l_{21} + l_{32}l_{22} & l_{31}^2 + l_{32}^2 + l_{33}^2 \end{bmatrix} \quad (127)$$

이를 통해 일대일로 비교하면 \mathbf{L} 의 원소는 다음과 같이 구할 수 있다.

$$\begin{aligned} l_{11} &= \sqrt{a_{11}} \quad \cdots \text{ up to sign} \\ l_{21} &= a_{21}/l_{11} \\ l_{31} &= a_{31}/l_{11} \\ l_{21} &= \sqrt{a_{22} - l_{21}^2} \\ l_{32} &= (a_{32} - l_{31}l_{21})/l_{22} \\ l_{33} &= \sqrt{a_{33} - l_{31}^2 - l_{32}^2} \end{aligned} \quad (128)$$

이를 임의의 행렬에 대해 일반화하여 표현하면 다음과 같다.

$$\begin{aligned} l_{ii} &= \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2} \\ l_{ij} &= \frac{1}{l_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik}l_{jk} \right) \end{aligned} \quad (129)$$

7.3 LDLT decomposition

Cholesky 분해에서 L 행렬의 대각 성분을 1로 만들기 위해 중앙에 대각행렬 D 를 별도로 분해하는 방법을 LDLT 분해라고 한다. 따라서 모든 cholesky 행렬은 LDLT 행렬로 분해할 수 있다.

$$\mathbf{A} = \mathbf{LL}^T = \mathbf{L}'\mathbf{DL}'^T \quad (130)$$

7.4 QR decomposition

QR 분해는 $\mathbf{Ax} = \mathbf{b}$ 시스템에서 행렬 \mathbf{A} 를 직교(orthogonal) 행렬 \mathbf{Q} 와 상삼각(upper-triangle) 행렬 \mathbf{R} 의 곱으로 분해하는 방법을 말한다.

$$\mathbf{A} = \mathbf{QR} \quad (131)$$

\mathbf{Q} 가 직교 행렬이므로 $\mathbf{QQ}^T = \mathbf{I}$ 의 성질을 지닌다. 일반적으로 QR 분해는 LU 분해보다 느리지만 최소제곱법(least squares) 문제를 풀 때 효율적이어서 자주 사용된다.

7.4.1 Detailed explanation

임의의 3×3 행렬 \mathbf{A} 가 주어졌을 때 이를 열벡터로 표현하면 다음과 같다. 자세한 내용은 [[6]]를 참고하면 된다.

$$\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3] \quad (132)$$

해당 행렬에 gram-schmidt 직교화를 수행하면 임의의 직교행렬 \mathbf{Q} 를 만들 수 있다.

$$\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3] \quad (133)$$

이 때, gram-schmidt 직교화의 특성 상 \mathbf{q}_1 은 첫번째 열벡터와 동일한 단위 벡터이고 \mathbf{q}_2 는 \mathbf{q}_1 와 직교한 단위벡터이며 \mathbf{q}_3 는 $\mathbf{q}_1, \mathbf{q}_2$ 와 직교한 단위벡터이다. 이를 통해 \mathbf{a}_i 를 구할 수 있다.

$$\begin{aligned} \mathbf{a}_1 &= \mathbf{a}_1^T \mathbf{q}_1 \cdot \mathbf{q}_1 \\ \mathbf{a}_2 &= \mathbf{a}_2^T \mathbf{q}_1 \cdot \mathbf{q}_1 + \mathbf{a}_2^T \mathbf{q}_2 \cdot \mathbf{q}_2 \\ \mathbf{a}_3 &= \mathbf{a}_3^T \mathbf{q}_1 \cdot \mathbf{q}_1 + \mathbf{a}_3^T \mathbf{q}_2 \cdot \mathbf{q}_2 + \mathbf{a}_3^T \mathbf{q}_3 \cdot \mathbf{q}_3 \end{aligned} \quad (134)$$

이를 행렬 형태로 표현하면 다음과 같은 상삼각 \mathbf{R} 행렬을 얻을 수 있다.

$$\begin{aligned} [\mathbf{a}_1 \quad \mathbf{a}_2 \quad \mathbf{a}_3] &= [\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3] \begin{bmatrix} \mathbf{a}_1^T \mathbf{q}_1 & \mathbf{a}_2^T \mathbf{q}_1 & \mathbf{a}_3^T \mathbf{q}_1 \\ & \mathbf{a}_2^T \mathbf{q}_2 & \mathbf{a}_3^T \mathbf{q}_2 \\ & & \mathbf{a}_3^T \mathbf{q}_3 \end{bmatrix} \\ \mathbf{A} &= \mathbf{QR} \\ \mathbb{R}^{3 \times 3} &= \mathbb{R}^{3 \times 3} \mathbb{R}^{3 \times 3} \end{aligned} \quad (135)$$

임의의 직사각 행렬에 대해서도 QR 분해를 수행할 수 있다. 만약 5×3 행렬 \mathbf{A} 가 주어졌을 때 이는 다음과 같이 QR 분해된다.

$$\begin{aligned} [\mathbf{a}_1 \quad \mathbf{a}_2 \quad \mathbf{a}_3] &= [\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{q}_4, \mathbf{q}_5] \begin{bmatrix} \mathbf{a}_1^T \mathbf{q}_1 & \mathbf{a}_2^T \mathbf{q}_1 & \mathbf{a}_3^T \mathbf{q}_1 \\ & \mathbf{a}_2^T \mathbf{q}_2 & \mathbf{a}_3^T \mathbf{q}_2 \\ & & \mathbf{a}_3^T \mathbf{q}_3 \end{bmatrix} \\ \mathbf{A} &= \mathbf{QR} \\ \mathbb{R}^{5 \times 3} &= \mathbb{R}^{5 \times 5} \mathbb{R}^{5 \times 3} \end{aligned} \quad (136)$$

$\mathbf{q}_4, \mathbf{q}_5$ 벡터는 곱셈에 의해 0이 되어서 실제 \mathbf{A} 행렬에는 관여하지 않는다.

7.4.2 QR decomposition on least squares problem

Over-determined 시스템 $\mathbf{Ax} = \mathbf{b}$ 가 주어졌을 때 이에 대한 최적해는 다음과 같이 최소제곱법을 통해 구할 수 있다.

$$\begin{aligned} \min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|_2^2 \\ \mathbf{x} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b} \end{aligned} \quad (137)$$

최소제곱법을 QR 분해를 통해 해석해보자. 임의의 직사각 행렬 \mathbf{A} 를 QR 분해해보면 다음과 같다.

$$\begin{aligned}\mathbf{A} &= \mathbf{QR} \\ &= [\mathbf{Q}_1 \quad \mathbf{Q}_2] \begin{bmatrix} \mathbf{R}_1 \\ \mathbf{0} \end{bmatrix}\end{aligned}\tag{138}$$

$\|\mathbf{Ax} - \mathbf{b}\|_2^2$ 를 QR 분해해보면 다음과 같다.

$$\begin{aligned}\|\mathbf{Ax} - \mathbf{b}\|_2^2 &= \|\mathbf{QRx} - \mathbf{b}\|_2^2 \\ &= \|\mathbf{Q}(\mathbf{Rx} - \mathbf{Q}^\top \mathbf{b})\|_2^2 \\ &= \|\mathbf{Rx} - \mathbf{Q}^\top \mathbf{b}\|_2^2 \\ &= \left\| \begin{bmatrix} \mathbf{R}_1 \\ \mathbf{0} \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathbf{Q}_1^\top \\ \mathbf{Q}_2^\top \end{bmatrix} \mathbf{b} \right\|_2^2\end{aligned}\tag{139}$$

위 식에서 세번째 줄은 $\|\mathbf{Q}(\cdot)\|_2^2 = (\cdot)^\top \mathbf{Q}^\top \mathbf{Q}(\cdot) = (\cdot)^\top (\cdot) = \|(\cdot)\|_2^2$ 을 통해 구할 수 있다. 네번째 줄은 QR 행렬을 블록 행렬 $\mathbf{Q}_i, \mathbf{R}_i$ 로 표현한 모습이다. 벡터의 제곱의 합은 선형성을 가지므로 위 식의 마지막 줄을 전개하면 다음과 같다.

$$\left\| \begin{bmatrix} \mathbf{R}_1 \\ \mathbf{0} \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathbf{Q}_1^\top \\ \mathbf{Q}_2^\top \end{bmatrix} \mathbf{b} \right\|_2^2 = \|\mathbf{R}_1 \mathbf{x} - \mathbf{Q}_1^\top \mathbf{b}\|_2^2 + \|\mathbf{Q}_2^\top \mathbf{b}\|_2^2\tag{140}$$

따라서 $\min_{\mathbf{x}} (\|\mathbf{R}_1 \mathbf{x} - \mathbf{Q}_1^\top \mathbf{b}\|_2^2 + \|\mathbf{Q}_2^\top \mathbf{b}\|_2^2)$ 를 최소화하는 \mathbf{x} 는 다음과 같이 구할 수 있다.

$$\mathbf{x} = \mathbf{R}^{-1} \mathbf{Q}^\top \mathbf{b}\tag{141}$$

이 때, 최소제곱법 식의 크기는 $\|\mathbf{Q}_2^\top \mathbf{b}\|_2^2$ 이다.

7.5 Eigen decomposition

정방행렬 $\mathbf{A} \in \mathbb{R}^{n \times n}$ 가 대각화 가능한 경우 다음과 같은 두 행렬 $\mathbf{V} \in \mathbb{R}^{n \times n}$ 과 대각행렬 $\mathbf{D} \in \mathbb{R}^{n \times n}$ 에 대하여 \mathbf{A} 를 다음과 같이 분해할 수 있다.

$$\mathbf{A} = \mathbf{VDV}^{-1}\tag{142}$$

이를 행렬 \mathbf{A} 에 대한 고유값 분해(eigen decomposition)라고 한다. 행렬 \mathbf{A} 가 대각화 가능하다는 의미는 행렬 \mathbf{A} 가 고유값 분해 가능하다는 말과 동치이다.

행렬 \mathbf{A} 가 대각화(고유값분해)되기 위해서는 역행렬이 존재하는 행렬 \mathbf{V} 가 존재해야 한다. 행렬 \mathbf{V} 가 역행렬이 존재하기 위해서는 \mathbf{V} 는 행렬 \mathbf{A} 와 같은 $\mathbb{R}^{n \times n}$ 크기의 정방행렬이어야 하고 n 개의 선형독립인 열벡터를 가지고 있어야 한다. 이 때, \mathbf{V} 의 각 열은 행렬 \mathbf{A} 의 고유벡터가 된다. 만약 행렬 \mathbf{V} 가 존재하는 경우 행렬 \mathbf{A} 는 대각화(고유값분해) 가능하다고 한다.

7.6 Singular value decomposition

$$\mathbf{A} = \mathbf{UDV}^\top = \begin{array}{c} m \\ \mathbf{U} \\ m \end{array} \quad \begin{array}{c} n \\ \mathbf{D} \\ m \end{array} \quad \begin{array}{c} n \\ \mathbf{V}^\top \\ n \end{array}$$

행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 이 주어졌을 때 특이값 분해(Singular Value Decomposition, SVD)는 다음과 같이 나타낼 수 있다.

$$\mathbf{A} = \mathbf{USV}^\top\tag{143}$$

이 때, $\mathbf{U} \in \mathbb{R}^{m \times m}$, $\mathbf{V} \in \mathbb{R}^{n \times n}$ 인 행렬이며 이들은 각 열이 Col A와 Row A에 의 정규직교기저벡터(Orthonormal Basis)로 구성되어 있다. $\Sigma \in \mathbb{R}^{m \times n}$ 은 대각행렬이며 대각 성분들이 $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{\min(m,n)}$ 특이값이며 큰 값부터 내림차순으로 정렬된 행렬이다.

7.6.1 Computing SVD

행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 에 대하여 \mathbf{AA}^\top 와 $\mathbf{A}^\top\mathbf{A}$ 는 다음과 같이 고유값분해할 수 있다.

$$\begin{aligned}\mathbf{AA}^\top &= \mathbf{U}\Sigma\mathbf{V}^\top\mathbf{V}\Sigma^\top\mathbf{U}^\top = \mathbf{U}\Sigma\Sigma^\top\mathbf{U}^\top = \mathbf{U}\Sigma^2\mathbf{U}^\top \\ \mathbf{A}^\top\mathbf{A} &= \mathbf{V}\Sigma^\top\mathbf{U}^\top\mathbf{U}\Sigma\mathbf{V}^\top = \mathbf{V}\Sigma^\top\Sigma\mathbf{U}^\top = \mathbf{V}\Sigma^2\mathbf{V}^\top\end{aligned}\quad (144)$$

이 때 계산되는 행렬 \mathbf{U}, \mathbf{V} 은 직교하는 고유벡터를 각 열의 성분으로 하는 행렬이며 대각행렬 Σ^2 의 각 성분은 항상 0보다 큰 양수의 값을 가진다. 그리고 \mathbf{AA}^\top 와 $\mathbf{A}^\top\mathbf{A}$ 를 통해 계산되는 Σ^2 의 값은 동일하다.

또한, 행렬 \mathbf{A} 에 대하여 $\mathbf{AA}^\top = \mathbf{A}^\top\mathbf{A} = \mathbf{S}$ 인 대칭행렬이 존재할 때 \mathbf{S} 가 Positive (Semi-)Definite한 경우

$$\begin{aligned}\mathbf{x}^\top\mathbf{AA}^\top\mathbf{x} &= (\mathbf{A}^\top\mathbf{x})^\top(\mathbf{A}^\top\mathbf{x}) = \|\mathbf{A}^\top\mathbf{x}\| \geq 0 \\ \mathbf{x}^\top\mathbf{A}^\top\mathbf{A}\mathbf{x} &= (\mathbf{Ax})^\top(\mathbf{Ax}) = \|\mathbf{Ax}\|^2 \geq 0\end{aligned}\quad (145)$$

즉, $\mathbf{AA}^\top = \mathbf{U}\Sigma^2\mathbf{U}^\top$ 와 $\mathbf{A}^\top\mathbf{A} = \mathbf{V}\Sigma^2\mathbf{V}^\top$ 에서 Σ^2 의 값은 항상 양수가 된다.

임의의 직각 행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 에 대하여 특이값 분해는 언제나 존재한다. $\mathbf{A} \in \mathbb{R}^{n \times n}$ 행렬의 경우 고유값 분해가 존재하지 않을 수 있지만 특이값 분해는 항상 존재한다. 대칭이면서 동시에 Positive Definite인 정방 행렬 $\mathbf{S} \in \mathbb{R}^{n \times n}$ 은 항상 고유값 분해값이 존재하며 이는 특이값 분해와 동일하다.

7.6.2 Range and nullspace of SVD

SVD는 다른 행렬 분해 방법들과 달리 \mathbf{A} 가 Singular하거나 Near-Singular한 경우에도 사용할 수 있는 방법이다. \mathbf{A} 가 Non-Singular한 경우 역행렬은 $\mathbf{A}^{-1} = \mathbf{V} \cdot \text{diag}(1/\sigma_j) \cdot \mathbf{U}^\top$ 와 같이 계산할 수 있다. 만약 \mathbf{A} 가 Singular한 경우 역행렬을 구할 때 몇몇 $\sigma_j = 0$ 이 되는데 이 때 $1/\sigma_j \Rightarrow 0$ 으로 설정함으로써 역행렬을 구할 수 있다. 특이값 σ_j 와 관련하여 SVD 분해는 다음과 같은 성질을 갖는다.

$\sigma_j \neq 0$ 일 때 이와 상응하는 \mathbf{U} 의 column들을 \mathbf{A} 행렬의 Orthogonal set of basis vector of Range라고 한다. $\sigma_j = 0$ 일 때 이와 상응하는 \mathbf{V} 의 column들을 \mathbf{A} 의 Orthogonal set of basis vector of Null Space라고 한다. 0이 아닌 특이값 $\sigma_j \neq 0$ 의 개수는 곧 행렬 \mathbf{A} 의 rank와 같다.

7.6.3 SVD on under-determined system

\mathbf{A} 가 Singular이면서 동시에 \mathbf{b} 가 Range 안에 포함되는 경우 선형시스템은 다수의 해를 가진다. 이런 경우 $\mathbf{Ax} = \mathbf{b}$ 에서 $\|\mathbf{x}\|^2$ 가 최소가 되는 해를 구할 수 있다.

$$\begin{aligned}\min_{\mathbf{x}} \|\mathbf{x}\|^2 \\ \mathbf{x} = \mathbf{V} \cdot \text{diag}(1/\sigma_j) \cdot \mathbf{U}^\top \cdot \mathbf{b}\end{aligned}\quad (146)$$

7.6.4 SVD on over-determined system

\mathbf{A} 가 Singular이면서 동시에 \mathbf{b} 가 Range에 존재하지 않는 경우 선형시스템은 해가 존재하지 않는다. 이런 경우 $\|\mathbf{Ax} - \mathbf{b}\|$ 가 최소가 되는 근사해를 구할 수 있다.

$$\begin{aligned}\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\| \\ \mathbf{x} = \mathbf{V} \cdot \text{diag}(1/\sigma_j) \cdot \mathbf{U}^\top \cdot \mathbf{b}\end{aligned}\quad (147)$$

7.7 Pseudo inverse

Pseudo Inverse는 선형시스템에서 행렬 \mathbf{A} 가 정방행렬이 아닐 경우 임의로 역행렬을 구하는 방법을 말한다. 이 때, 선형시스템은 일반적으로 full column rank 또는 full row rank일 때 pseudo inverse를 적용할 수 있다. Full rank가 아닌 행렬에 대한 pseudo inverse는 추후 섹션에서 설명한다.

7.7.1 Pseudo inverse on under-determined system

under-determined 시스템의 경우 \mathbf{A} 는 full row rank가 되고 pseudo inverse는 다음과 같이 정의된다. lagrange multiplier λ 를 포함하여 최적화 문제를 정의하면 다음과 같다.

$$\min_{\mathbf{x}} \|\mathbf{x}\|^2 + \lambda^\top(\mathbf{b} - \mathbf{Ax}) \quad (148)$$

이를 미분 후 0으로 만드는 값을 찾으면 다음과 같다.

$$2\mathbf{x} - \mathbf{A}^\top\lambda = 0 \quad (149)$$

\mathbf{A} 가 정방행렬이 아니기 때문에 바로 해를 구할 수 없으므로 양변의 왼쪽에 \mathbf{A} 를 곱한다.

$$2\mathbf{Ax} - \mathbf{AA}^\top \lambda = 0 \quad (150)$$

이 때, $\mathbf{Ax} = \mathbf{b}$ 이므로 다음과 같은 식이 유도된다.

$$2\mathbf{b} = \mathbf{AA}^\top \lambda \quad (151)$$

$$\lambda = 2(\mathbf{AA}^\top)^{-1}\mathbf{b} \quad (152)$$

따라서 \mathbf{x} 는 다음과 같이 구할 수 있다.

$$\mathbf{x} = \mathbf{A}^\top(\mathbf{AA}^\top)^{-1}\mathbf{b} \quad (153)$$

$$\boxed{\mathbf{A}^\dagger = \mathbf{A}^\top(\mathbf{AA}^\top)^{-1} \quad \dots \text{ for under-determined system}} \quad (154)$$

즉, under-determined 시스템에서 pseudo inverse는 오른쪽에 곱해지게 되며 이를 right pseudo inverse라고 부르기도 한다. (유도 과정이 불확실함. 확실하게 알게되는대로 업데이트 예정. 자세한 유도 과정을 아시는 분은 연락주시면 감사하겠습니다.)

$$\begin{aligned} \mathbf{AA}^\dagger \mathbf{x} &= \mathbf{A}^\dagger \mathbf{b} \\ \mathbf{x} &= \mathbf{A}^\dagger \mathbf{b} \\ \mathbf{x} &= \mathbf{A}^\top(\mathbf{AA}^\top)^{-1}\mathbf{b} \quad \dots \text{ for under-determined system} \end{aligned} \quad (155)$$

7.7.2 Pseudo inverse on over-determined system

over-determined 시스템의 경우 \mathbf{A} 는 full column rank가 되고 pseudo inverse는 다음과 같이 정의된다. 우선, over-determined 시스템에 대한 최적화 문제는 다음과 같이 최소제곱법 문제가 된다.

$$\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|^2 = \min_{\mathbf{x}} \|\mathbf{b} - \mathbf{Ax}\|^2 = \min_{\mathbf{x}} (\mathbf{b} - \mathbf{Ax})^\top (\mathbf{b} - \mathbf{Ax}) \quad (156)$$

이를 전개하면 다음과 같다.

$$\min_{\mathbf{x}} \mathbf{b}^\top \mathbf{b} - \mathbf{b}^\top \mathbf{Ax} - \mathbf{x}^\top \mathbf{A}^\top \mathbf{b} + \mathbf{x}^\top \mathbf{A}^\top \mathbf{Ax} \quad (157)$$

위 문제를 풀기 위해 \mathbf{x} 에 대해 미분하면 다음과 같은 식이 얻어진다.

$$-(\mathbf{b}^\top \mathbf{A})^\top - (\mathbf{A}^\top \mathbf{b}) + 2\mathbf{A}^\top \mathbf{Ax} = 0 \quad (158)$$

따라서 \mathbf{x} 는 다음과 같이 구할 수 있다.

$$\mathbf{x} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b} \quad (159)$$

$$\boxed{\mathbf{A}^\dagger = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \quad \dots \text{ for over-determined system}} \quad (160)$$

즉, over-determined 시스템에서 pseudo inverse는 왼쪽에 곱해지게 되며 이를 left pseudo inverse라고 부르기도 한다.

$$\begin{aligned} \mathbf{A}^\dagger \mathbf{Ax} &= \mathbf{A}^\dagger \mathbf{b} \\ \mathbf{x} &= \mathbf{A}^\dagger \mathbf{b} \\ \mathbf{x} &= (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b} \quad \dots \text{ for over-determined system} \end{aligned} \quad (161)$$

7.7.3 SVD of pseudo inverse

선형 시스템 $\mathbf{Ax} = \mathbf{b}$ 이 주어졌을 때 임의의 직사각형 행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 와 pseudo inverse \mathbf{A}^\dagger 는 SVD 분해를 통해 다음과 같이 나타낼 수 있다.

$$\begin{aligned} \mathbf{A} &= \mathbf{U} \Sigma \mathbf{V}^\top \\ \mathbf{A}^\dagger &= \mathbf{V} \Sigma^\dagger \mathbf{U}^\top \end{aligned} \quad (162)$$

- $\mathbf{U} \in \mathbb{R}^{m \times m}$

- $\Sigma = \text{diag}(\sigma_1, \sigma_2, \sigma_3, \dots) \in \mathbb{R}^{m \times n}$

- $\Sigma^\dagger = \text{diag}(1/\sigma_1, 1/\sigma_2, 1/\sigma_3, \dots) \in \mathbb{R}^{n \times m}$

- $\mathbf{V} \in \mathbb{R}^{n \times n}$

7.7.4 Full column rank case

임의의 직사각형 행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 가 full column rank를 가지면 pseudo inverse는 $\mathbf{A}^\dagger \mathbf{A}$ 와 같이 왼쪽에 곱해진다. 이를 SVD 분해하여 표현하면 다음과 같다.

$$\mathbf{A}^\dagger \mathbf{A} = \mathbf{V} \Sigma^\dagger \mathbf{U}^\top \mathbf{U} \Sigma \mathbf{V}^\top = \mathbf{I}_n \quad (163)$$

또한, 앞서 구한 (160)에서 $\mathbf{A}^\dagger = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top$ 를 풀어쓰면 다음과 같다.

$$\begin{aligned} \mathbf{A}^\dagger &= (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \\ &= (\mathbf{V} \Sigma^\top \mathbf{U}^\top \mathbf{U} \Sigma \mathbf{V}^\top)^{-1} \mathbf{V} \Sigma^\top \mathbf{U}^\top \\ &= \mathbf{V} \Sigma^{-2} \Sigma \mathbf{U}^\top \quad \dots \Sigma^\top = \Sigma \\ &= \mathbf{V} \Sigma^\dagger \mathbf{U}^\top \quad \dots \Sigma^\dagger = \Sigma^{-1} \end{aligned} \quad (164)$$

이는 앞서 정의한 (162)와 동일하다.

7.7.5 Full row rank case

임의의 직사각형 행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 가 full row rank를 가지면 pseudo inverse는 $\mathbf{A} \mathbf{A}^\dagger$ 와 같이 오른쪽에 곱해진다. 이를 SVD 분해하여 표현하면 다음과 같다.

$$\mathbf{A} \mathbf{A}^\dagger = \mathbf{U} \Sigma \mathbf{V}^\top \mathbf{V} \Sigma^\dagger \mathbf{U}^\top = \mathbf{I}_m \quad (165)$$

또한, 앞서 구한 (154)에서 $\mathbf{A}^\dagger = \mathbf{A}^\top (\mathbf{A} \mathbf{A}^\top)^{-1}$ 를 풀어쓰면 다음과 같다.

$$\begin{aligned} \mathbf{A}^\dagger &= \mathbf{A}^\top (\mathbf{A} \mathbf{A}^\top)^{-1} \\ &= \mathbf{V} \Sigma^\top \mathbf{U}^\top (\mathbf{U} \Sigma \mathbf{V}^\top \mathbf{V} \Sigma^\dagger \mathbf{U}^\top)^{-1} \\ &= \mathbf{V} \Sigma \Sigma^{-2} \mathbf{U}^\top \quad \dots \Sigma^\top = \Sigma \\ &= \mathbf{V} \Sigma^\dagger \mathbf{U}^\top \quad \dots \Sigma^\dagger = \Sigma^{-1} \end{aligned} \quad (166)$$

이는 앞서 정의한 (162)와 동일하다.

7.7.6 Rank deficient case

만약 임의의 직사각형 행렬 \mathbf{A} 이 full rank가 아닐 경우 pseudo inverse는 다음과 같이 나타낼 수 있다. 예를 들어 $\mathbf{A} \in \mathbb{R}^{3 \times 4}$ 일 때 이를 SVD 분해하면 다음과 같다.

$$\begin{aligned} \mathbf{A} &= \mathbf{U} \Sigma \mathbf{V}^\top \\ &= \mathbf{U} \begin{bmatrix} \sigma_1 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{V}^\top \end{aligned} \quad (167)$$

Pseudo inverse \mathbf{A}^\dagger 를 SVD 분해하면 다음과 같다.

$$\begin{aligned} \mathbf{A}^\dagger &= \mathbf{V} \Sigma^\dagger \mathbf{U}^\top \\ &= \mathbf{V} \begin{bmatrix} 1/\sigma_1 & 0 & 0 \\ 0 & 1/\sigma_2 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{U}^\top \end{aligned} \quad (168)$$

따라서 오른쪽 pseudo inverse $\mathbf{A} \mathbf{A}^\dagger$ 는 다음과 같이 전개할 수 있다.

$$\begin{aligned} \mathbf{A} \mathbf{A}^\dagger &= \mathbf{U} \Sigma \mathbf{V}^\top \mathbf{V} \Sigma^\dagger \mathbf{U}^\top \\ &= \mathbf{U} \begin{bmatrix} 1 & & \\ & 1 & \\ & & 0 \end{bmatrix} \mathbf{U}^\top \\ &= \mathbf{u}_1 \mathbf{u}_1^\top + \mathbf{u}_2 \mathbf{u}_2^\top \end{aligned} \quad (169)$$

만약 \mathbf{A} 가 full rank인 경우 \mathbf{AA}^\dagger 는 다음과 같다.

$$\begin{aligned}\mathbf{AA}^\dagger &= \mathbf{I}_3 \\ &= \mathbf{u}_1\mathbf{u}_1^\top + \mathbf{u}_2\mathbf{u}_2^\top + \mathbf{u}_3\mathbf{u}_3^\top\end{aligned}\tag{170}$$

따라서 rank deficient 케이스의 경우 \mathbf{AA}^\dagger 는 마지막 $\mathbf{u}_3\mathbf{u}_3^\top$ 이 없는 pseudo inverse가 구해진다. 이는 항등행렬 \mathbf{I}_3 와 유사한 값을 갖지만 동일하지는 않은 행렬이다.

다음으로 왼쪽 pseudo inverse $\mathbf{A}^\dagger\mathbf{A}$ 는 다음과 같이 전개할 수 있다.

$$\begin{aligned}\mathbf{A}^\dagger\mathbf{A} &= \mathbf{V}\Sigma^\dagger\mathbf{U}^\top\mathbf{U}\Sigma\mathbf{V}^\top \\ &= \mathbf{V}\begin{bmatrix} 1 & & \\ & 1 & \\ & & 0 \end{bmatrix}\mathbf{V}^\top \\ &= \mathbf{v}_1\mathbf{v}_1^\top + \mathbf{v}_2\mathbf{v}_2^\top\end{aligned}\tag{171}$$

만약 \mathbf{A} 가 full rank인 경우 $\mathbf{A}^\dagger\mathbf{A}$ 는 다음과 같다.

$$\begin{aligned}\mathbf{A}^\dagger\mathbf{A} &= \mathbf{I}_4 \\ &= \mathbf{v}_1\mathbf{v}_1^\top + \mathbf{v}_2\mathbf{v}_2^\top + \mathbf{v}_3\mathbf{v}_3^\top + \mathbf{v}_4\mathbf{v}_4^\top\end{aligned}\tag{172}$$

따라서 rank deficient 케이스의 경우 $\mathbf{A}^\dagger\mathbf{A}$ 는 마지막 $\mathbf{v}_3\mathbf{v}_3^\top + \mathbf{v}_4\mathbf{v}_4^\top$ 이 없는 pseudo inverse가 구해진다. 이는 항등행렬 \mathbf{I}_4 와 유사한 값을 갖지만 동일하지는 않은 행렬이다. 이는 \mathbf{AA}^\dagger 를 통해 구한 행렬보다 덜 항등행렬에 근접하다.

따라서 임의의 non-full rank를 가지는 직사각형 행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 이 주어졌을 때, $m < n$ 인 경우 \mathbf{AA}^\dagger 를 수행하는 것이 더 항등행렬에 근접한 pseudo inverse를 수행할 수 있고 $m > n$ 인 경우 $\mathbf{A}^\dagger\mathbf{A}$ 를 수행하는 것이 더 항등행렬에 근접한 pseudo inverse를 수행할 수 있다.

7.7.7 QR decomposition of pseudo inverse when singular case

간혹 $\mathbf{A}^\dagger\mathbf{A}$ 가 singular하거나 near-singular한 경우 QR 분해를 사용하여 pseudo inverse를 구한다.

$$\begin{aligned}\mathbf{x} &= \mathbf{A}^\dagger\mathbf{b} \\ &= (\mathbf{A}^\dagger\mathbf{A})^{-1}\mathbf{A}^\dagger\mathbf{b} \\ &= (\mathbf{R}^\top\mathbf{Q}^\top\mathbf{Q}\mathbf{R})^{-1}\mathbf{R}^\top\mathbf{Q}^\top\mathbf{b} \\ &= (\mathbf{R}^\top\mathbf{R})^{-1}\mathbf{R}^\top\mathbf{Q}^\top\mathbf{b} \\ &= \mathbf{R}^{-1}\mathbf{Q}^\top\mathbf{b}\end{aligned}\tag{173}$$

위 식은 (141)와 동일하다.

7.8 Sherman-Morrison formula

역행렬이 존재하는 임의의 행렬 \mathbf{A} 에 대해 rank 1 업데이트를 하는 방법을 Sherman-Morrison 공식이라고 한다. 해당 공식에 대한 보다 자세한 내용은 [[6]]를 참조하면 된다.

$$(\mathbf{A} + \mathbf{uv}^\top)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{uv}^\top\mathbf{A}^{-1}}{1 + \mathbf{v}^\top\mathbf{A}^{-1}\mathbf{u}}\tag{174}$$

위 식에서 $(1 + \mathbf{v}^\top\mathbf{A}^{-1}\mathbf{u}) \neq 0$ 와 $(\mathbf{A} + \mathbf{uv}^\top)^{-1}$ 이 역행렬이 존재하는 조건은 동치이다. 이 때, \mathbf{u}, \mathbf{v} 는 임의의 두 벡터를 의미하며 이를 \mathbf{uv}^\top 와 같이 곱하면 항상 rank 1 행렬이 생성된다.

$$\mathbf{uv}^\top = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \begin{bmatrix} v_1 & v_2 \end{bmatrix} = \begin{bmatrix} v_1 & v_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \dots \text{ linearly dependent} = \text{rank 1}\tag{175}$$

7.8.1 Recursive least squares

Sherman-Morrison 공식은 데이터가 계속 추가되는 최소제곱법 문제에 사용하면 연산량을 적게 소모하면서 효율적으로 역행렬을 업데이트할 수 있다. 다음과 같은 선형 시스템 $\mathbf{Ax} = \mathbf{b}$ 가 주어졌다고 하자. $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{x} \in \mathbb{R}^{n \times 1}$, $\mathbf{b} \in \mathbb{R}^{m \times 1}$ 일 때 이를 풀어서 쓰면 아래와 같다.

$$\begin{aligned} \mathbf{Ax} &= \mathbf{b} \\ \begin{bmatrix} \mathbf{a}_1^\top \\ \mathbf{a}_2^\top \\ \vdots \\ \mathbf{a}_m^\top \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} &= \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \end{aligned} \quad (176)$$

선형시스템의 최소제곱법의 해는 다음과 같다.

$$\mathbf{x} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b} \quad (177)$$

만약 $m+1$ 번째 데이터 \mathbf{a}_{m+1}^\top 이 입력되면 이에 맞게 최적해를 업데이트해줘야 한다. 표현의 편의를 위해 $m+1$ 번째 데이터를 \mathbf{a} 로 표현하면 다음과 같다.

$$\begin{aligned} \mathbf{x} &= \left([\mathbf{A}^\top \quad \mathbf{a}] \begin{bmatrix} \mathbf{A}^\top \\ \mathbf{a} \end{bmatrix} \right)^{-1} [\mathbf{A}^\top \quad \mathbf{a}] \begin{bmatrix} \mathbf{b} \\ b \end{bmatrix} \\ &= (\mathbf{A}^\top \mathbf{A} + \mathbf{a} \mathbf{a}^\top)^{-1} (\mathbf{A}^\top \mathbf{b} + \mathbf{a} b_{m+1}) \end{aligned} \quad (178)$$

이 때, 앞 부분 $(\mathbf{A}^\top \mathbf{A} + \mathbf{a} \mathbf{a}^\top)^{-1}$ 에 Sherman-Morrisson 공식 (174)을 적용하면 연산량을 적게 소모하면서 효율적으로 최적해를 업데이트할 수 있다. 이는 다음과 같이 전개 후 치환하여 간결하게 나타낼 수 있다.

$$\begin{aligned} (\mathbf{A}^\top \mathbf{A} + \mathbf{a} \mathbf{a}^\top)^{-1} &= (\mathbf{A}^\top \mathbf{A})^{-1} - \frac{(\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{a} \mathbf{a}^\top (\mathbf{A}^\top \mathbf{A})^{-1}}{1 + \mathbf{a}^\top (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{a}} \\ &= \mathbf{P} - \frac{\mathbf{P} \mathbf{a} \mathbf{a}^\top \mathbf{P}}{1 + \mathbf{a}^\top \mathbf{P} \mathbf{a}} \\ &= \mathbf{P}_a \end{aligned} \quad (179)$$

- $(\mathbf{A}^\top \mathbf{A})^{-1} = \mathbf{P}$ 표현의 편의를 위해 치환한다

위 치환한 식을 기반으로 (178)를 전개하면 다음과 같다.

$$\begin{aligned} (\mathbf{A}^\top \mathbf{A} + \mathbf{a} \mathbf{a}^\top)^{-1} (\mathbf{A}^\top \mathbf{b} + \mathbf{a} b_{m+1}) &= \left(\mathbf{P} - \frac{\mathbf{P} \mathbf{a} \mathbf{a}^\top \mathbf{P}}{1 + \mathbf{a}^\top \mathbf{P} \mathbf{a}} \right) (\mathbf{A}^\top \mathbf{b} + \mathbf{a} b_{m+1}) \\ &= \mathbf{P} \mathbf{A}^\top \mathbf{b} + \frac{\mathbf{P} \mathbf{a} \mathbf{a}^\top \mathbf{P}}{1 + \mathbf{a}^\top \mathbf{P} \mathbf{a}} \mathbf{A}^\top \mathbf{b} + \mathbf{P}_a \mathbf{a} b \\ &= \mathbf{x} - \frac{\mathbf{P} \mathbf{a} \mathbf{a}^\top \mathbf{P}}{1 + \mathbf{a}^\top \mathbf{P} \mathbf{a}} \mathbf{A}^\top \mathbf{b} + \mathbf{P}_a \mathbf{a} b \\ &= \mathbf{x} - \left(\frac{\mathbf{P} \mathbf{a}}{1 + \mathbf{a}^\top \mathbf{P} \mathbf{a}} \right) \mathbf{a}^\top \mathbf{x} + \mathbf{P}_a \mathbf{a} b \\ &= \mathbf{x} - (\mathbf{P}_a \mathbf{a}) \mathbf{a}^\top \mathbf{x} + \mathbf{P}_a \mathbf{a} b \\ &= \mathbf{x} + \mathbf{P}_a \mathbf{a} (b - \mathbf{a}^\top \mathbf{x}) \end{aligned} \quad (180)$$

- $\mathbf{P} \mathbf{A}^\top \mathbf{b} = \mathbf{x}$

위 식에서 5번째 줄은 $\mathbf{P}_a \mathbf{a}$ 를 전개한 후 분모를 통분하여 정리함으로써 유도할 수 있다. 따라서 데이터가 증가했을 때 새로운 최적해는 이전 최적해 식으로부터 아래와 같이 업데이트된다. 이를 recursive least squares(RLS)라고 한다.

$$\boxed{\mathbf{x} \leftarrow \mathbf{x} + \mathbf{P}_a \mathbf{a} (b - \mathbf{a}^\top \mathbf{x})} \quad (181)$$

7.9 Matrix inversion lemma

Matrix inversion lemma는 역행렬 변환 공식을 의미하며 선형 시스템을 다룰 때 자주 쓰이는 트릭 중 하나이다. 이는 Sherman-Morrison-Woodbury 공식이라고도 불린다. Matrix inversion lemma는 다음과 같이 정의된다. Lemma에 대한 보다 자세한 내용은 [[6]]를 참조하면 된다.

$$\boxed{(\mathbf{A} + \mathbf{UCV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1} \mathbf{U} (\mathbf{C}^{-1} + \mathbf{V} \mathbf{A}^{-1} \mathbf{U})^{-1} \mathbf{V} \mathbf{A}^{-1}} \quad (182)$$

- $\mathbf{A} \in \mathbb{R}^{n \times n}$ - $\mathbf{U} \in \mathbb{R}^{n \times k}$ - $\mathbf{C} \in \mathbb{R}^{k \times k}$ - $\mathbf{V} \in \mathbb{R}^{k \times n}$ - $\mathbf{A}, \mathbf{C}, \mathbf{C}^{-1} + \mathbf{V}\mathbf{A}^{-1}\mathbf{U}$ is invertible

7.9.1 Derivation of matrix inversion lemma

Matrix inversion lemma를 유도하기 위해 4개의 블록 행렬로 구성된 \mathbf{M} 가 주어졌다고 하자.

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \quad (183)$$

7.9.2 LDU decomposition

다음으로 \mathbf{M} 를 LDU 분해하려고 한다. 아래와 같이 \mathbf{C} 를 소거하기 위한 행렬을 곱해서 LU 행렬을 만들 수 있다.

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{CA}^{-1} & \mathbf{I} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{CA}^{-1} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{CA}^{-1} & \mathbf{I} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{0} & \mathbf{D} - \mathbf{CA}^{-1}\mathbf{B} \end{bmatrix} \quad (184)$$

이 때, $\mathbf{D} - \mathbf{CA}^{-1}\mathbf{B}$ 를 \mathbf{A} 의 schur complement (\mathbf{M}/\mathbf{A})라고 한다. 다음으로 대각 행렬 성분만 남기기 위해 아래와 같이 오른쪽에 행렬을 전개하면 LDU 분해가 마무리된다.

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{CA}^{-1} & \mathbf{I} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{0} & \mathbf{D} - \mathbf{CA}^{-1}\mathbf{B} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{CA}^{-1} & \mathbf{I} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{D} - \mathbf{CA}^{-1}\mathbf{B} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{A}^{-1}\mathbf{B} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \quad (185)$$

\mathbf{M}^{-1} 은 다음과 같이 LDU 행렬을 사용하여 전개할 수 있다.

$$\begin{aligned} \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}^{-1} &= \begin{bmatrix} \mathbf{I} & -\mathbf{A}^{-1}\mathbf{B} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A}^{-1} & \mathbf{0} \\ \mathbf{0} & (\mathbf{D} - \mathbf{CA}^{-1}\mathbf{B})^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{C}^{-1}\mathbf{A} & \mathbf{I} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}(\mathbf{D} - \mathbf{CA}^{-1}\mathbf{B})^{-1}\mathbf{CA}^{-1} & -\mathbf{A}^{-1}\mathbf{B}(\mathbf{D} - \mathbf{CA}^{-1}\mathbf{B})^{-1} \\ -(\mathbf{D} - \mathbf{CA}^{-1}\mathbf{B})^{-1}\mathbf{CA}^{-1} & (\mathbf{D} - \mathbf{CA}^{-1}\mathbf{B})^{-1} \end{bmatrix} \end{aligned} \quad (186)$$

7.9.3 UDL decomposition

행렬 \mathbf{M} LDU 뿐만 아니라 UDL로도 분해될 수 있다. 아래와 같이 \mathbf{B} 를 소거하기 위한 행렬을 곱해서 UL 행렬을 만들 수 있다.

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{D}^{-1}\mathbf{C} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{D}^{-1}\mathbf{C} & \mathbf{I} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{A} - \mathbf{BD}^{-1}\mathbf{C} & \mathbf{B} \\ \mathbf{0} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{D}^{-1}\mathbf{C} & \mathbf{I} \end{bmatrix}^{-1} \quad (187)$$

이 때, $\mathbf{A} - \mathbf{BD}^{-1}\mathbf{C}$ 를 \mathbf{D} 의 schur complement (\mathbf{M}/\mathbf{D})라고 한다. 다음으로 대각 행렬 성분만 남기기 위해 아래와 같이 왼쪽에 행렬을 전개하면 UDL 분해가 마무리된다.

$$\begin{bmatrix} \mathbf{A} - \mathbf{BD}^{-1}\mathbf{C} & \mathbf{B} \\ \mathbf{0} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{D}^{-1}\mathbf{C} & \mathbf{I} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{I} & \mathbf{BD}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A} - \mathbf{BD}^{-1}\mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{D}^{-1}\mathbf{C} & \mathbf{I} \end{bmatrix}^{-1} \quad (188)$$

\mathbf{M}^{-1} 은 다음과 같이 UDL 행렬을 사용하여 전개할 수 있다.

$$\begin{aligned} \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}^{-1} &= \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{D}^{-1}\mathbf{C} & \mathbf{I} \end{bmatrix} \begin{bmatrix} (\mathbf{A} - \mathbf{BD}^{-1}\mathbf{C})^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{BD}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \\ &= \begin{bmatrix} (\mathbf{A} - \mathbf{BD}^{-1}\mathbf{C})^{-1} & -(\mathbf{A} - \mathbf{BD}^{-1}\mathbf{C})^{-1}\mathbf{BD} \\ -\mathbf{D}^{-1}\mathbf{C}(\mathbf{A} - \mathbf{BD}^{-1}\mathbf{C})^{-1} & \mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{C}(\mathbf{A} - \mathbf{BD}^{-1}\mathbf{C})^{-1}\mathbf{BD} \end{bmatrix} \end{aligned} \quad (189)$$

7.9.4 Back to matrix inversion lemma

앞서 구한 (186), (189)는 분해 방법만 달랐을 뿐 모든 원소는 서로 같아야 한다. 따라서 첫번째 원소를 비교해보면 다음과 같다.

$$(\mathbf{A} - \mathbf{BD}^{-1}\mathbf{C})^{-1} = \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}(\mathbf{D} - \mathbf{CA}^{-1}\mathbf{B})^{-1}\mathbf{CA}^{-1} \quad (190)$$

해당 식에서 아래와 같이 기호만 변경해주면 matrix inversion lemma 식 (182)가 된다.

$$\begin{aligned}\mathbf{B} &\rightarrow \mathbf{U} \\ \mathbf{C} &\rightarrow \mathbf{V} \\ \mathbf{D}^{-1} &\rightarrow -\mathbf{C} \\ \therefore (\mathbf{A} + \mathbf{UCV})^{-1} &= \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{C}^{-1} + \mathbf{VA}^{-1}\mathbf{U})^{-1}\mathbf{VA}^{-1}\end{aligned}\tag{191}$$

또한 (186), (189)의 두번째 원소를 비교하면 다음과 같다. 해당 식도 자주 사용되는 행렬 변환 트릭 중 하나이다.

$$-\mathbf{A}^{-1}\mathbf{B}(\mathbf{D} - \mathbf{CA}^{-1}\mathbf{B})^{-1} = -(\mathbf{A} - \mathbf{BD}^{-1}\mathbf{C})^{-1}\mathbf{BD}\tag{192}$$

지금까지 소개한 matrix inversion lemma 행렬 변환 트릭은 칼만 필터(kalman filter)의 공식을 유도할 때 종종 사용되며 이외에도 많은 공학 분야에서 사용된다.

8 Reference

- [1] (Lecture) edwith 인공지능을 위한 선형대수, 주재길 교수
- [2] (Book) Kay, Steven M. Fundamentals of statistical signal processing: estimation theory. Prentice-Hall, Inc., 1993.
- [3] (Blog) 다크프로그래머 - Gradient, Jacobian 행렬, Hessian 행렬, Laplacian
- [4] (Blog) [행렬대수학] 행렬식(Determinant) 1 - 행렬식의 개념
- [5] (Pdf) Pseudo Inverse 유도 과정
- [6] (Youtube) Matrix Inversion Lemma 강의 영상 - 혁펜하임

9 Revision log

- 1st: 2020-05-15
- 2nd: 2020-06-21
- 3rd: 2023-01-21
- 4th: 2023-01-31
- 5th: 2024-02-24